SIEMENS

TRANSDATA

9749, 9750, 9752 Data Display Terminals Programmer's Interface

User's Guide

Revision November 1984

Order No. U1593-J1-Z76-2-7600 Printed in the Federal Republic of Germany 630 AG 7850.4 (790)	Order No.: U1593-J1-Z76-2-7600	

TRANSDATA

9749, 9750, 9752 Data Display Terminals Programmer's Interface

User's Guide

9750 Data Display Terminal with Ergolift (option)

Edition December 1983
Revision November 1984

Order No. U1593-J-Z76-1-7600
Printed in the Federal Republic of Germany
2620 AG 12830.5 (3280)

Order No. U1593-J1-Z76-2-7600
Printed in the Federal Republic of Germany
630 AG 7850.4 (790)

Performance features may be added, modified or omitted in the course of product development. Other information in this publication is subject to the same conditions.

Siemens Aktiengesellschaft

PREFACE

Contents

This manual provides information on:

- TRANSDATA network configurations,
- the functions offered by the data display terminals,
- how these functions can be physically and logically utilized and programmed,
- how the data display terminals are supported by the software components in the communication computer and host computer,
- data exchange with printers that can be connected to the data display terminals,
- how the P keys can be loaded using the software product "PLUS".

An appendix at the end of the manual contains important tables for programmers.

Readership

The contents of the manual are aimed primarily at the experienced programmer who

- needs to program output to data display terminals and printers
- wishes to interpret and evaluate correctly entries made at the data display terminals.

The manual may also be useful as a course document, offering novices a quick survey of the physical and logical programming of data display terminal functions.

Requirements

The user of this manual should have a knowledge of programming and of the software components involved.

In order to enable the full range of terminal functions to be utilized physically as well as logically, the text includes references to the latest manuals and other publications.

Your views count

To help us continue to improve our publications, please send us your comments, requests and suggestions using the pink reply forms provided.

Editorial Department D ST PM 2

LIST OF AMENDMENTS 1

The following amendments have been made since publication of the
Edition December 1983,
and incorporated in the
Revision November 1984:

Chapter 5, section 7.5 and chapter 9 have been added to the manual.

CONTENTS

Contents

BS2000

System Messages — Reference Manual
Systems Standard — Reference Manual

DMS Tape Processing — Reference Manual
DMS Disk Processing — Reference Manual
Executive Macros — Reference Manual
LEASY — Reference Manual
RFA — Reference Manual

LMS — Reference Manual
Laser Printer — Reference Manual
ARCHIVE — Reference Manual
Utility Routines — Reference Manual

EDT — Reference Manual
EDOR — Reference Manual
SORT — Reference Manual
PERCON — Reference Manual

Control System Command Language — Reference Manual
Job Variables — Reference Manual
Interactive Debugging Aid — Reference Manual

EASP + — Beschreibung

Introductory Guide for System Users — User's Guide
Introductory Guide for System Operators — User's Guide

TRANSDATA

Systems Standards — Reference Manual
Directions for the Terminal User — User's Guide
9749, 9750, 9752 Data Display Terminals Programmer's Interface — User's Guide
9001 Printer Programmer's Interface — User's Guide
9003 Printer Programmer's Interface — User's Guide
9004 Printer Programmer's Interface — User's Guide

DCAM Program Interfaces — Reference Manual
DCAM COBOL Calls — User's Guide
DCAM Macros — User's Guide

UTM Program Interfaces — Reference Manual
UTM KDCS COBOL Calls — User's Guide
UTM Generation and Administration — User's Guide
UTM-RPC1 UTM for RPG Users — User's Guide

TIAM — User's Guide

FHS — User's Guide
IFG for FHS — User's Guide

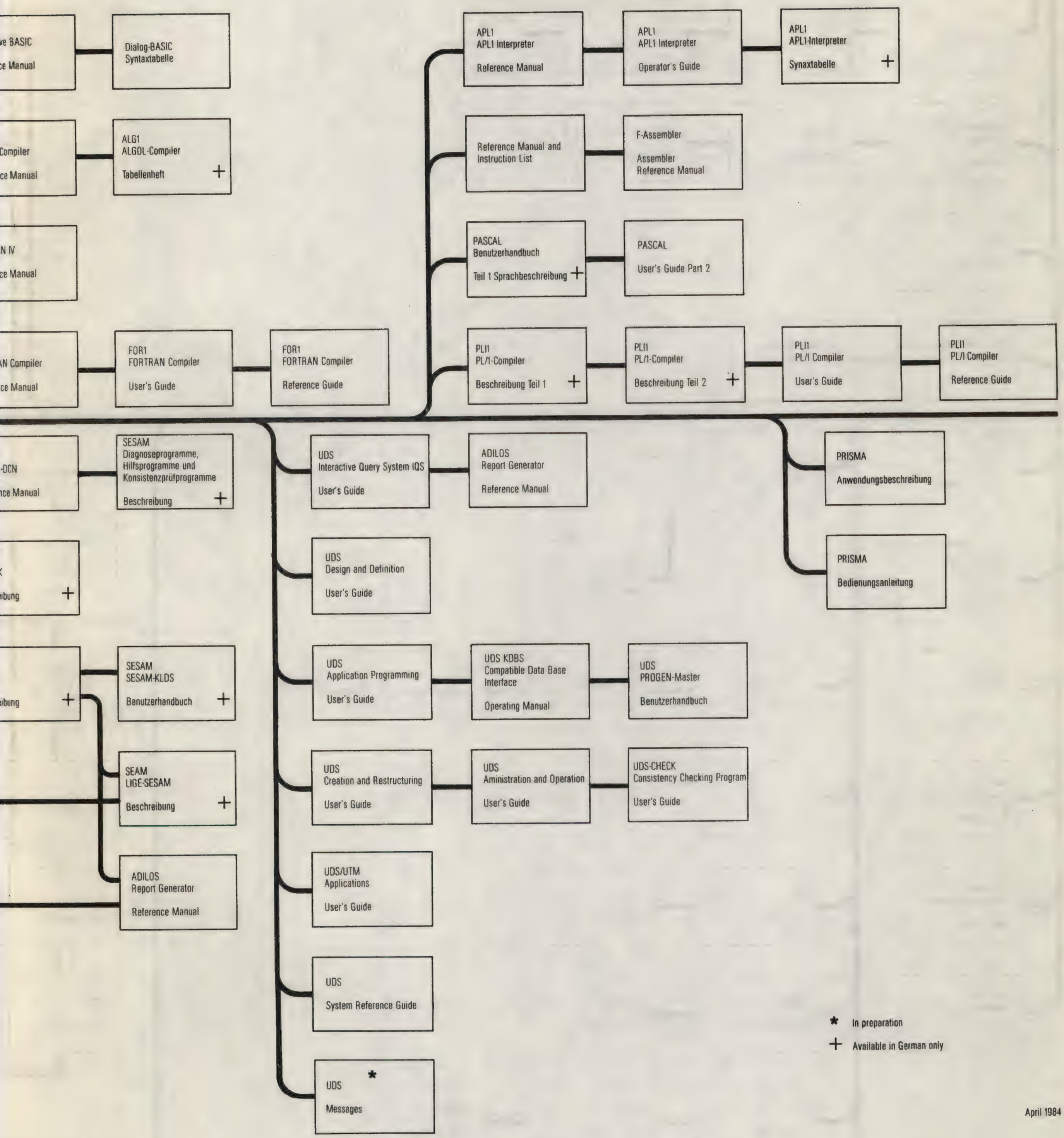
ASMUS — User's Guide
ASMUS2 + — Benutzerhandbuch

RCOM — User's Guide
RCOM-S — Reference Manual
RCOM FTP — Reference Manual

File Transfer — User's Guide

Connection to Packet Switching Networks (X.25, X.29) — Reference Manual

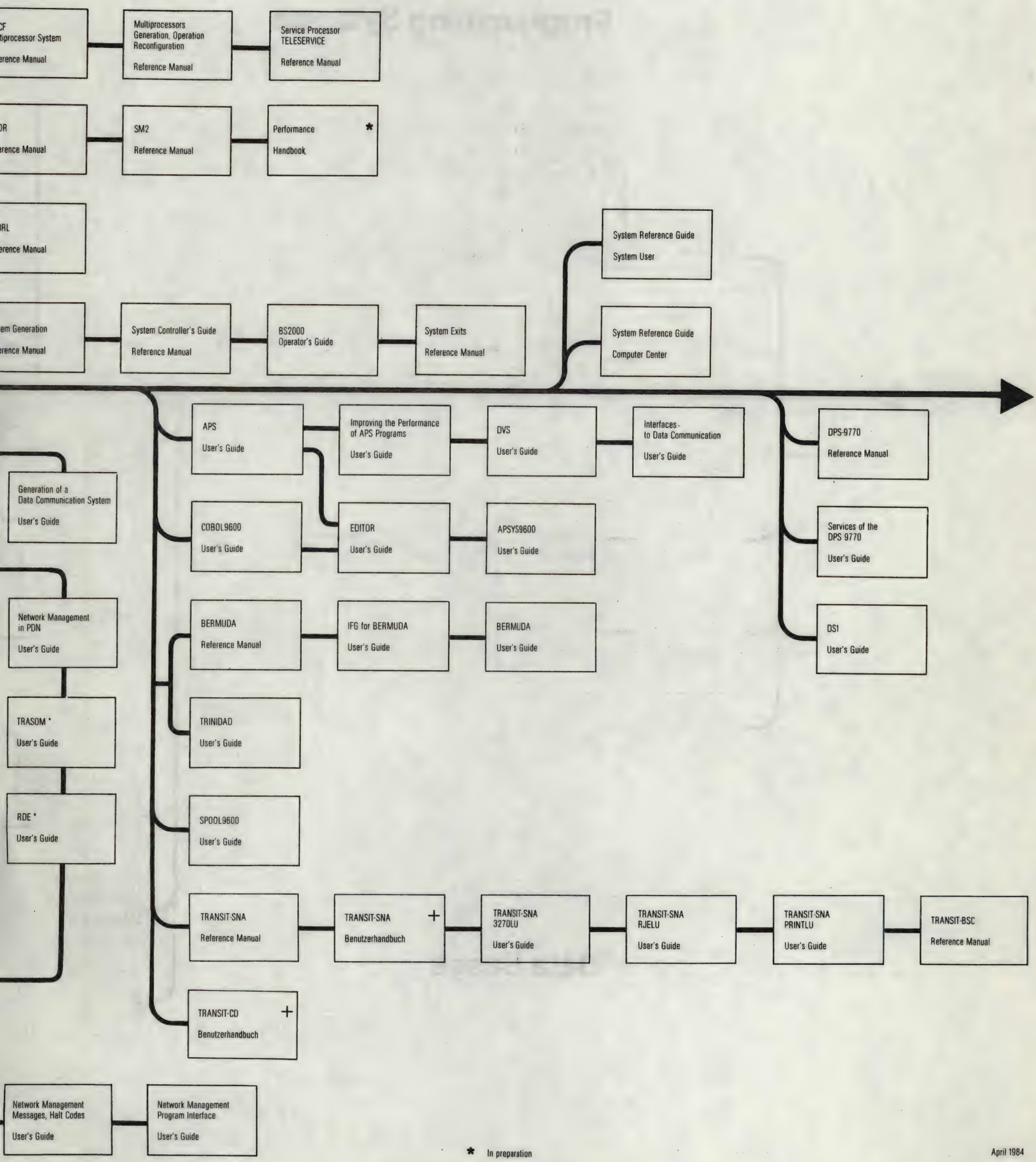
Generation of Preferred Configurations — User's Guide
Network Management in BS2000 — User's Guide
NEOSP — User's Guide
TDA * — User's Guide
ADS + — Benutzerhandbuch
RBAS — User's Guide
OMNIS — User's Guide


ve BASIC
ce Manual
Dialog-BASIC
Syntaxtabelle
Compiler
ce Manual
ALG1
ALGOL-Compiler
Tabellenheft +
N IV
ce Manual
AN Compiler
ce Manual
FOR1
FORTRAN Compiler
User's Guide
FOR1
FORTRAN Compiler
Reference Guide
APL1
APL1 Interpreter
Reference Manual
APL1
APL1 Interpreter
Operator's Guide
APL1
APL1-Interpreter
Synaxtabelle +
Reference Manual and
Instruction List
F-Assembler
Assembler
Reference Manual
PASCAL
Benutzerhandbuch
Teil 1 Sprachbeschreibung +
PASCAL
User's Guide Part 2
PLI1
PL/1-Compiler
Beschreibung Teil 1 +
PLI1
PL/1-Compiler
Beschreibung Teil 2 +
PLI1
PL/I Compiler
User's Guide
PLI1
PL/I Compiler
Reference Guide
-DCN
nce Manual
SESAM
Diagnoseprogramme,
Hilfsprogramme und
Konsistenzprüfprogramme
Beschreibung +
UDS
Interactive Query System IQS
User's Guide
ADILOS
Report Generator
Reference Manual
PRISMA
Anwendungsbeschreibung
PRISMA
Bedienungsanleitung
ibung +
UDS
Design and Definition
User's Guide
ibung +
SESAM
SESAM-KLDS
Benutzerhandbuch +
UDS
Application Programming
User's Guide
UDS KDBS
Compatible Data Base
Interface
Operating Manual
UDS
PROGEN-Master
Benutzerhandbuch
SEAM
LIGE-SESAM
Beschreibung +
UDS
Creation and Restructuring
User's Guide
UDS
Aministration and Operation
User's Guide
UDS-CHECK
Consistency Checking Program
User's Guide
ADILOS
Report Generator
Reference Manual
UDS/UTM
Applications
User's Guide
UDS
System Reference Guide
UDS *
Messages
* In preparation
+ Available in German only

April 1984

Publications Plan for

- BS2000
- TRANSDATA
- Programming Systems
- Data Bases

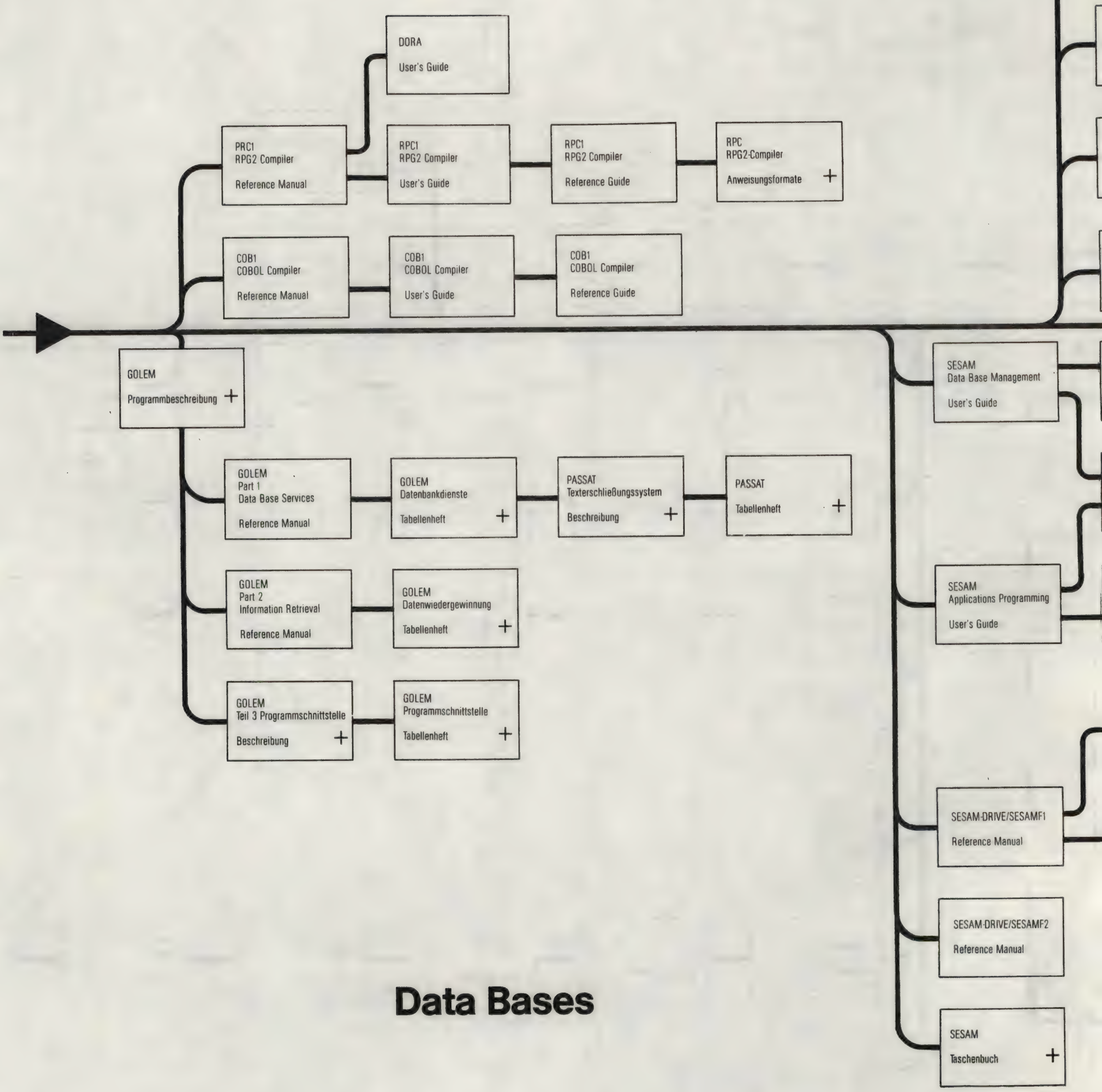


* In preparation

\+ Available in German only

April 1984

Programming Systems



Data Bases

1 INTRODUCTION

Technological development supports the trend towards the decentralization of data processing.
Siemens has embodied this trend in the hardware and software components of its systems and in TRANSDATA.

1.1 TRANSDATA CONCEPT

TRANSDATA, a data processing system and communications system all in one (hardware and software), is both concept and product.
The user of TRANSDATA distinguishes between the modes of application available in teleprocessing.
The following diagram illustrates the concept of the five TRANSDATA application types.

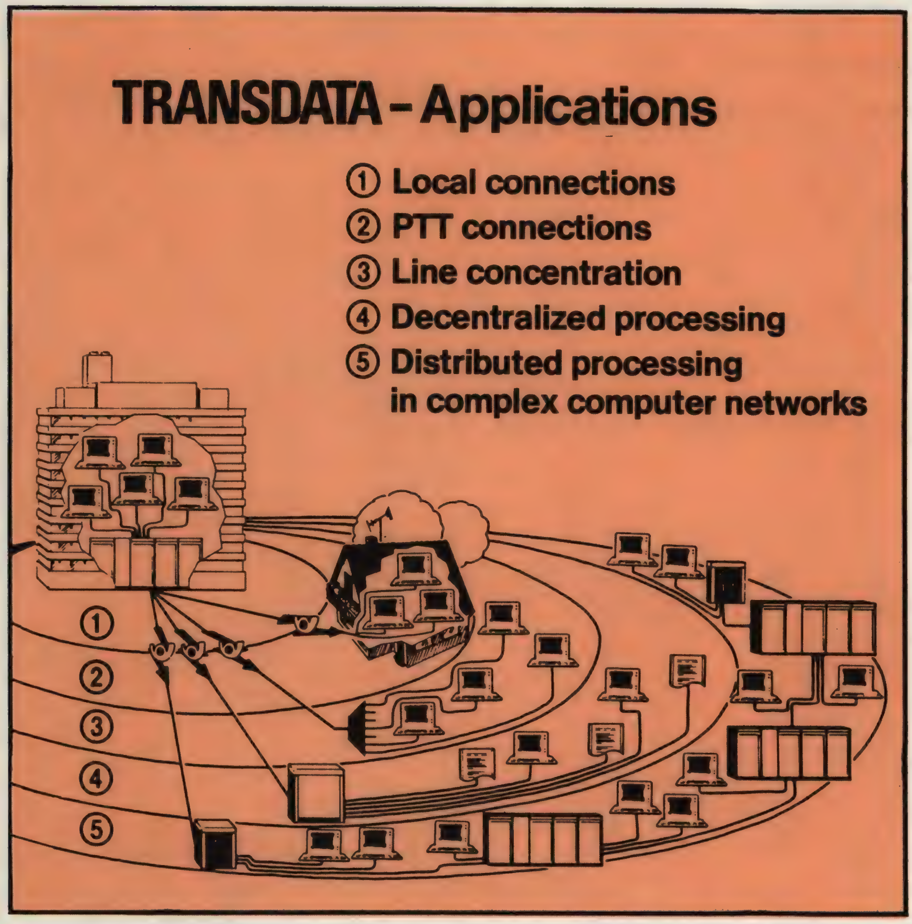


Fig. 1-1 TRANSDATA applications

1.1.1 TRANSDATA Hardware

A large number of host computers, together with an extensive range of TRANSDATA hardware products that interoperate with one another enable the design of diverse network configurations.

TRANSDATA hardware products are

- (standard) front-end processor (FEP), compact front-end processor (CFEP),
- remote front-end processor (FEPR),
- terminal computer (TC) with local peripherals and TRANSDATA 960 System products,
- data terminals,
- additional communication devices
  - cluster controllers (local, remote),
  - interface expanders (IE),
  - medium-speed concentrators (MSC),
  - protocol/converter concentrators (PCC).

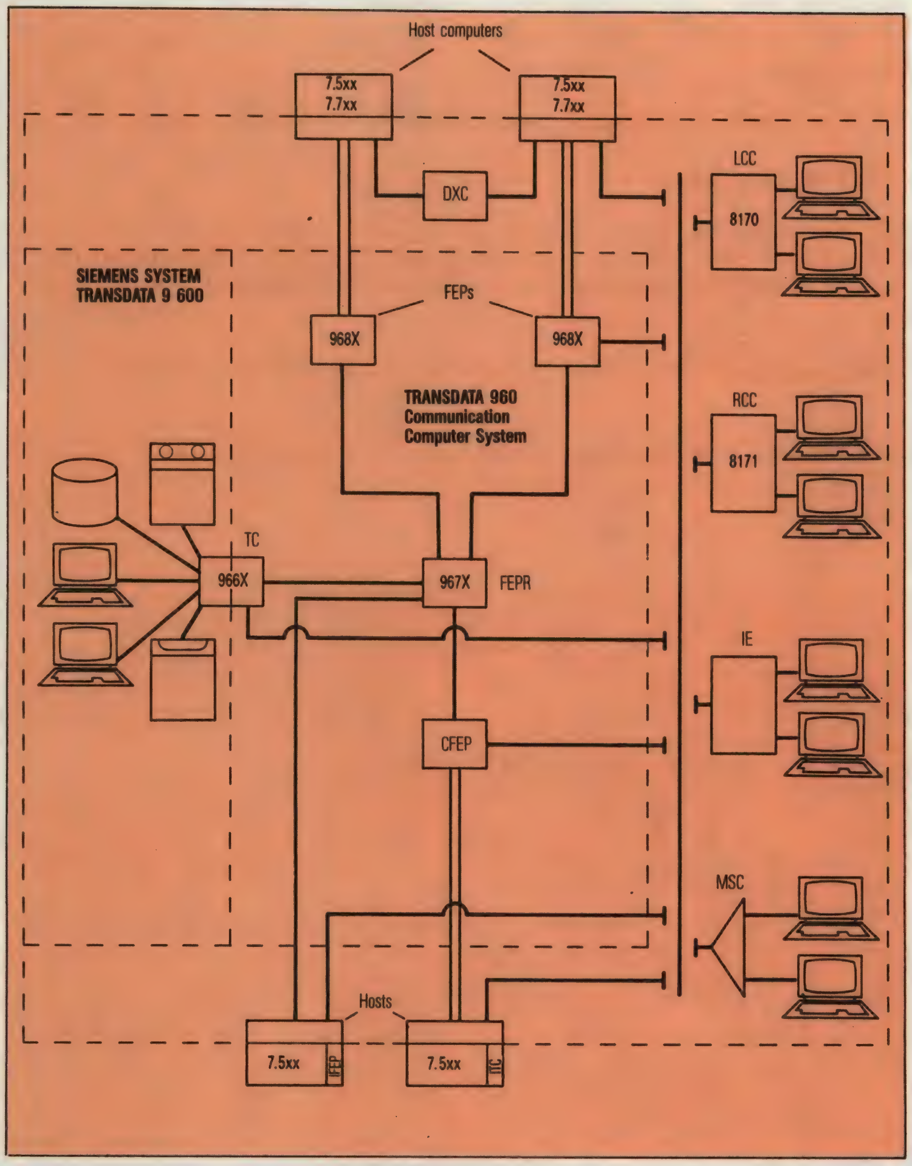


Fig. 1-2 TRANSDATA hardware

1.1.2 TRANSDATA Software

TRANSDATA software comprises

- PDN (Program System for Teleprocessing and Network Control) and
- DCM (Data Communication Methods),

which interoperate and complement each other. They are responsible for the transmission of data between the sender and receiver in the communication system, control and supervise communication and edit the data in a form suitable for the user:

- All TRANSDATA 960 System communication computers run on the PDN operating system.
- All the host computers use the communication access system DCM.

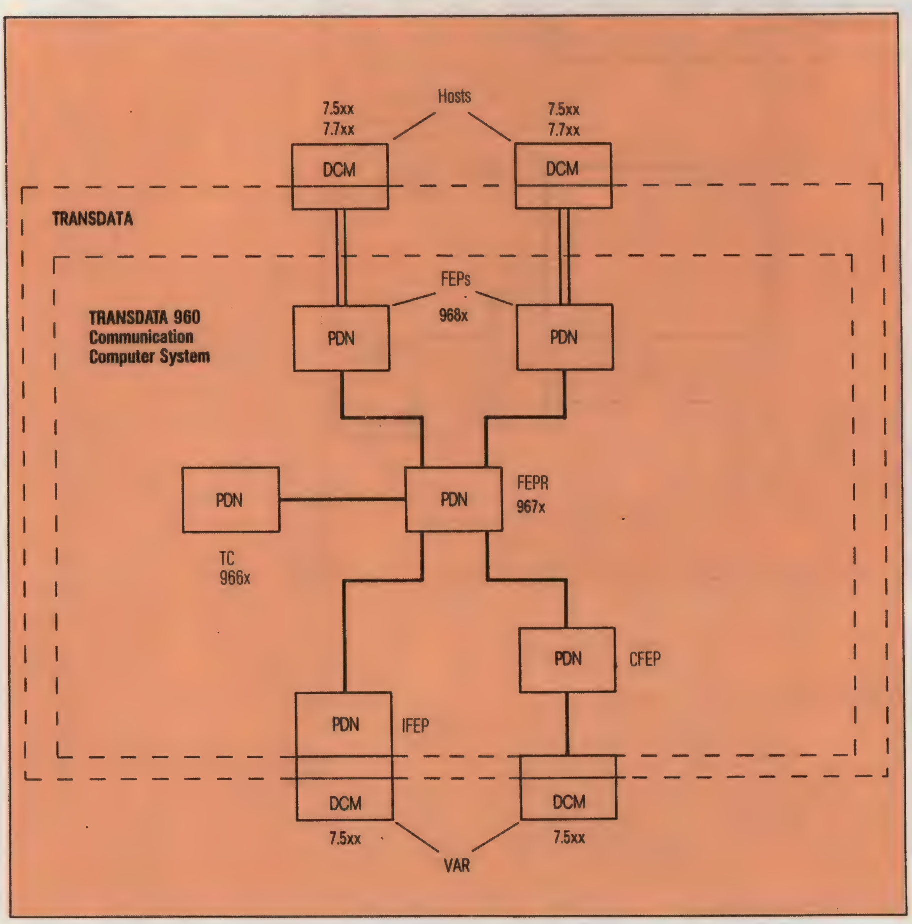


Fig. 1-3 TRANSDATA software

1.2 DATA DISPLAY TERMINAL CONCEPT

1.2.1 Basic Requirements

Data display terminals are appraised according to performance, cost-effectiveness and ergonomic design. The 9749, 9750 and 9752 Data Display Terminals meet the following basic demands:

- The functions and peripheral devices are ideally suited to interactive mode and data entry applications.
- Their design complies with acknowledged ergonomic criteria as well as with relevant standards and safety regulations.
- Flexible design of a wide range of data networks.
- Incorporation into existing applications and adaptability to given organizational requirements is ensured.

1.2.2 Compatibility

The versions of the 9749 and 9750 Terminals are compatible with the functions of the 8160 and 8162 Terminals in all hardware and software configurations.
The 9752 Terminal is compatible with the 9749 and 9750 Terminals in relation to the system. The central software can, without modification, interoperate simultaneously with 9749, 9750 and 9752 Data Display Terminals.

1.2.3 Operating Modes

An increasing number of new areas in working life are being won over and supported by applications of the dialog mode, with the data display terminals being the most important factor.
Interactive (dialog) mode on the data display terminals may be either

- unformatted or
- formatted (e.g. BERMUDA).

The following diagram gives an overview of the possible operating modes.

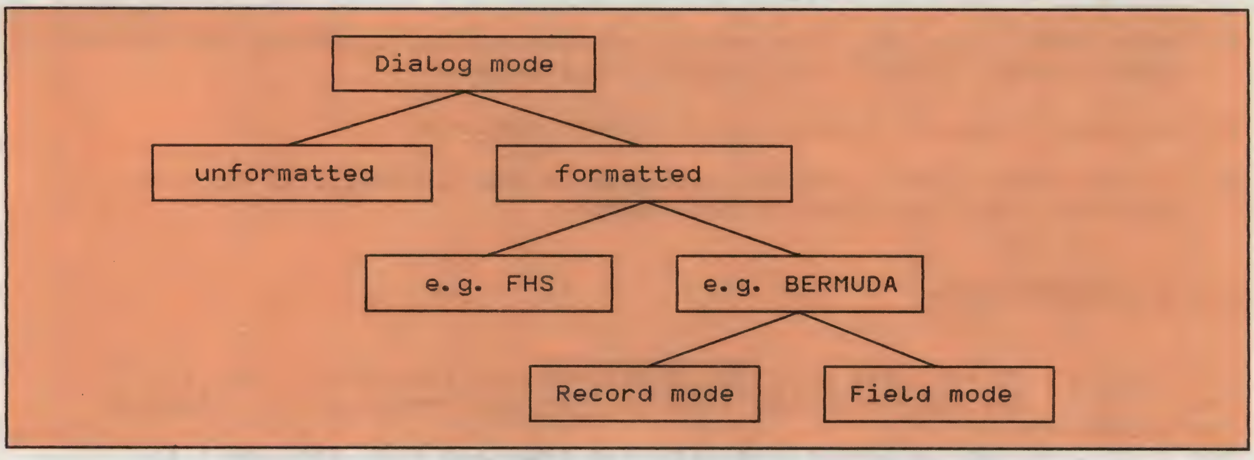


Figure 1-4 Operating modes

Unformatted Mode

When a data display terminal user works with the communication application $DIALOG, the data exchange with the operating system (commands, messages) is unformatted. Utility routines are also unformatted.
Application programs can work with data display terminals in unformatted mode.

Terminal-to-host transmission
In unformatted mode, the user has the whole screen at his disposal for data entry. A message is sent to the host computer by depressing one of the data transmission keys (e.g. DÜ1).

Host-to-terminal transmission
In unformatted mode, the messages sent from the host computer to the data display terminal will appear on the screen in the following manner:

- always from the beginning of the next new line

  and in roll-up mode

- from the beginning of the last line.

Formatted Mode

During formatted operation the terminal screen is divided into

- protected fields and
- unprotected fields.

Application programs can work with data display terminals in formatted mode.

<u>Terminal-to-host transmission</u>
Only the unprotected fields can be filled by the user, via the keyboard. After being filled or modified, the unprotected (variable) fields can be transmitted to the host computer. The transmission process is the same as for unformatted mode.

<u>Host-to-terminal transmission</u>
Both protected and unprotected fields are outputted during transmission from host to terminal.

- The <u>protected</u> fields cannot be modified via the terminal keyboard.
- The <u>unprotected</u> fields are modifiable and can be described via the keyboard with text which should be sent to the host computer.

Formatted Mode with BERMUDA

The data display terminal can operate in conjunction with the software service BERMUDA (User Service for Terminal Mask Support) in the following modes

- record mode or
- field mode.

The terminal computer activates the operating modes by loading the parameter ranges. The user cannot perform mode switchover.

<u>Record mode</u>
The terminal computer outputs a mask onto the screen. The mask contains operator prompts and variable fields. The user can fill in consecutively all the variable fields in the mask. He can position freely within the mask without affecting BERMUDA.
After the mask has been partially or completely filled, the user can initiate a send sequence to the terminal computer. BERMUDA then takes over the processing of the input data i.e. syntax checking, plausibility checking and field editing.

<u>Field mode</u>
In field mode, only one mask field at a time is released for the user, namely the current one, and the cursor is set to the first position in it. Every character entered is subject to syntax checking and plausibility checking by the data display terminal.
Transmission is initiated only after these checks are successfully completed. The field is edited by the display terminal and then sent to the terminal computer which, in its next output sequence, releases the next field for filling in by the user [13].

2 GENERAL DESCRIPTION OF THE DATA DISPLAY TERMINALS

2.1 COMPONENTS OF THE DATA DISPLAY TERMINALS

In the basic version, display terminals comprise two components:

- display terminal
- keyboard.

2.1.1 Versions of the Data Display Terminal

The screen, power supply and the data transmission unit are accommodated in a casing suitable for the display workstation. The following table shows the characteristics of the different versions of the 9749, 9750 and 9752 Data Display Terminals.

Features	Versions of the data display terminal											
	9749-1	9750-1	9750-2	9750-3	9750-4	9750-5	9750-7	9750-8	9750-9	9750-62	9752-1	9752-2
Standalone terminal	—	X	—	X	—	X	—	—	—	—	—	—
Cluster terminal	X	—	X	—	X	—	X	X	X	X	X	X
12-inch screen	—	X	X	—	—	—	—	—	—	—	—	—
14-inch screen	—	—	—	—	—	—	—	—	—	—	X	X
15-inch screen	X	—	—	X	X	X	X	X	X	X	—	—
Integrated printer connection	—	—	—	—	—	—	—	X	—	—	—	—
Integrated printer conn. poss.	X	X	X	X	X	X	—	—	X	X	X	X
No printer connection possible	—	—	—	—	—	—	X	—	—	—	—	—
Presetting memory	—	X	X	X	—	X	—	—	X	—	X	X
ROM	X	—	—	—	X	—	X	X	—	X	—	—
Interactive mode	X	X	X	X	X	X	X	X	—	X	X	X
Interactive mode and BERMUDA	—	—	W 2)	—	—	—	—	—	X	—	—	X
X.21 interface	—	—	—	—	—	X	—	—	—	—	—	—
Keyswitch retrofitting 1)	X	X	X	X	X	X	X	X	X	X	X	X

1) Keylock switch for user identification

2) Optional

Table 2-1 Versions of the data display terminal

2.1.2 Keyboard Versions

Different versions (4) of electronic keyboards are available which connect to the display terminal via a connecting cable.
Keyboard versions differ by their character assignments, which may be

- task-specific and/or
- language-specific with special national characters.

The following table shows the characteristics of the various keyboard versions.

Assignment	Keyboard versions			
	Interactive mode		Interactive mode + BERMUDA	
	Upper and lower case	Upper case only	Upper and lower case	Upper case only
International	97500-11	97500-21	97500-31	97500-41
German	-12	-22	-32	-42
Belgian	-13	--	-33	--
	-17	--	-37	--
Swedish	-14	--	-34	--
Danisch	-15	--	-35	--
French	-16	--	-36	--
Swiss	-18	--	-38	--

Table 2-2 Keyboard versions

2.2 OPTIONS

The following table shows which additional devices can be used together with the data display terminals.

Options	Attachable to 9749-X	9750-X	9752-X	Remark, Brief description
97507-X Integrated, manual badge reader	X	X	X	Manual badge reader for reading magnetically encoded ID cards (DIN 9781) where information coding is on track 2 in accordance with ISO 3554 (ABA standard) or SIPASS (switchable).
9006 OCR Scanner	X	X	X	Scanner for - OCR-A: Eurobanking (25 characters) Standard (27 characters). - OCR-B: Standard (24 characters).
97505-4 Ergolift	X	X	X	With Ergolift the terminal can be raised, lowered, swiveled and tilted.

Table 2-3 Options

2.3 PRINTER CONNECTION

A number of universal, high-performance printers (e.g. 9001 and 9003 Printers) are available for connection to the data display terminals.

The printers are connected to the terminals locally, via the SS81 or SS97 Interface.

The printers produce a hard copy of the screen contents.

The printers can also be controlled directly by the host computer via the transmission line (bypass mode); the terminal then acts as a printer terminal controller.

An independent print buffer in the display terminals enables interactive mode to run concurrently with printer output.

2.4 CONNECTING OPTIONS AND NETWORK SCHEMATIC OF DATA DISPLAY TERMINALS

Data display terminals can be operated as

- Stand-alone data display terminals:
  only 9750
- Clustered data display terminals:
  9749, 9750 and 9752.

2.4.1 Stand-Alone Terminals

Stand-alone terminals can be connected directly or via all current data communication equipment to the host computer:

- Public leased-circuit data network
- Switched telephone network
- Datex network (Datex-L network or Datex-P network in conjunction with the 9120 Protocol Converter/Concentrator together with synchronous data terminal equipment)

- Interface Expander (IE)
- Concentrator
- Communication computers of the TRANSDATA 960 System.

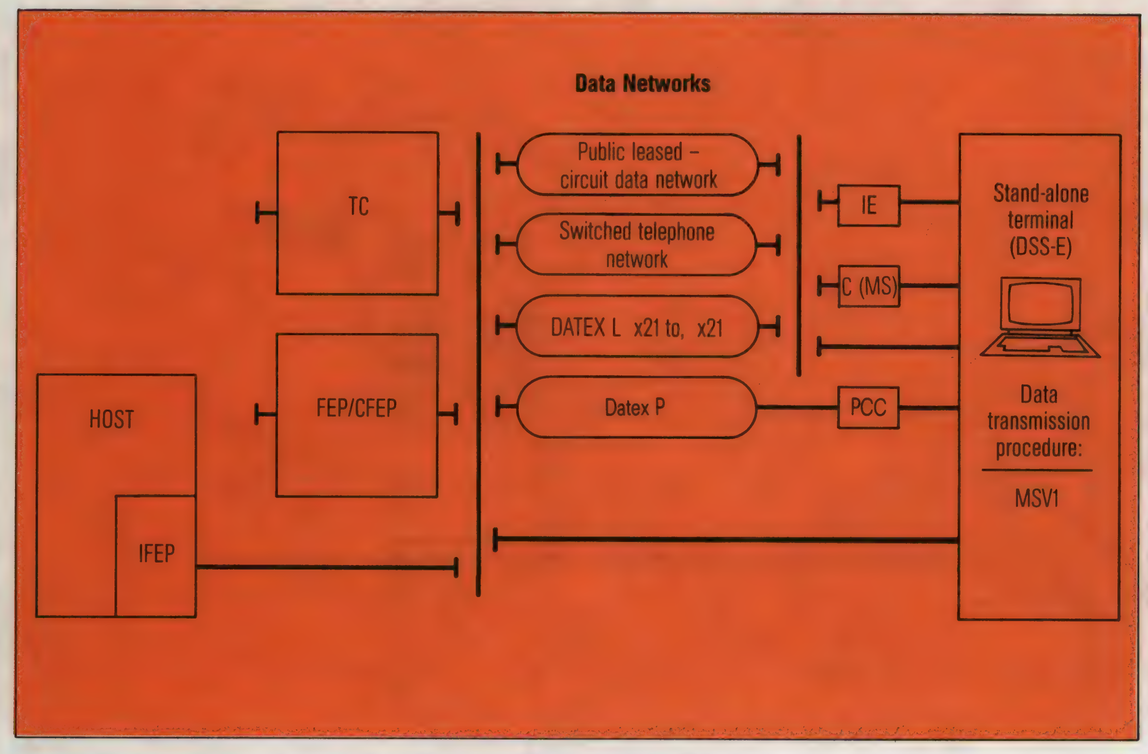


Fig. 2-1 Stand alone data display terminals

2.4.2 Clustered data display terminals

Clustered data display terminals are connected locally to

- Cluster Controllers,
  - 8170 Cluster Controller (local),
  - 8171 Cluster Controller (remote),
- Communication computers of the TRANSDATA 960 System via the BAM interface controller,
- Host computers (host and IFEP) via the Integrated Terminal Controller (ITC).

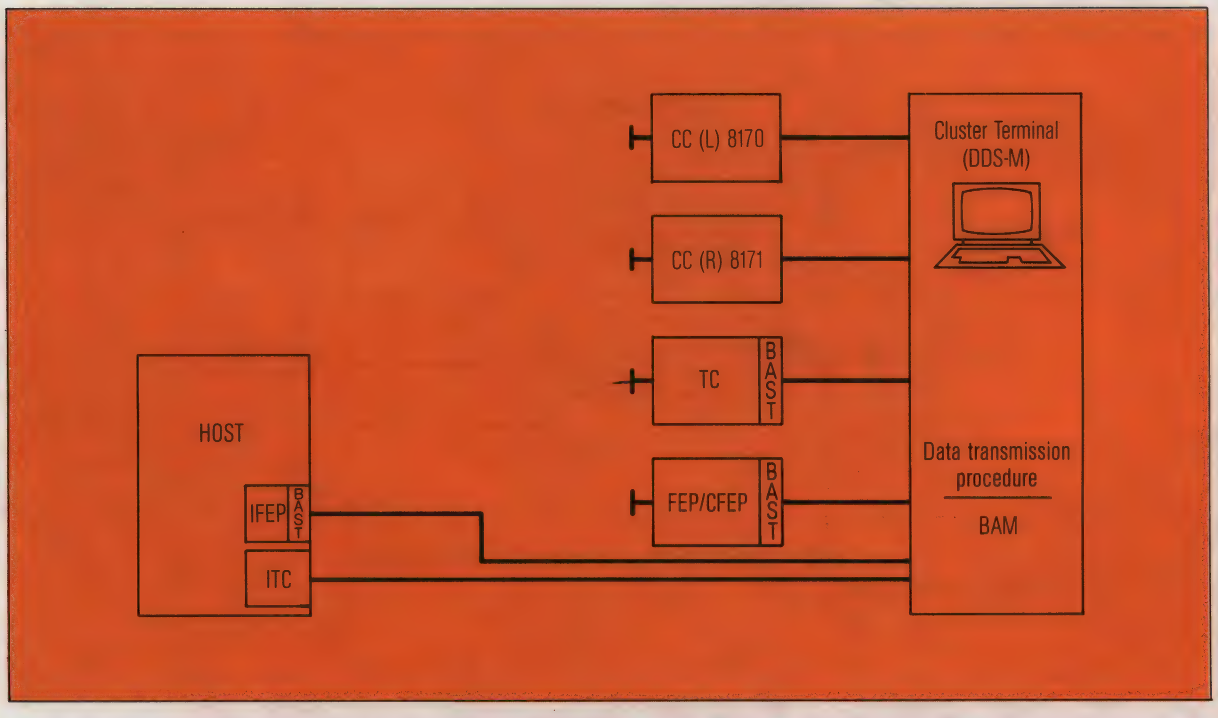


Fig. 2-2 Clustered data display terminals

2.5 EXAMPLES OF NETWORK CONFIGURATIONS

2

A comprehensive range of hardware products tailored to perform a wide variety of functions, permit the realization of network configurations to suit individual user requirements. The following examples can give, therefore, only an overview of where and how display terminals can be connected.

2.5.1 Connection to Cluster Controllers

Cluster controllers permit a number of data display terminals to be connected cost-effectively to a host computer. Data transmission facilities and lines are reduced in this way as opposed to the situation with standalone terminals.
Depending on the distance between the host computer and the data display terminals to be connected, either the 8170 or 8171 Cluster Controller is used.
Four terminals can connect to a cluster controller in the basic configuration and further expansion options permit a total of 32 to be connected.

Note

The connection of an 8171 Cluster Controller to a concentrator allows only 8 terminals to be connected.

8170 Cluster Controller

The 8170 Cluster Controller is employed in the immediate vicinity of the host computer (within a range of max. 30m) and is connected to the byte multiplex channel of the host computer.

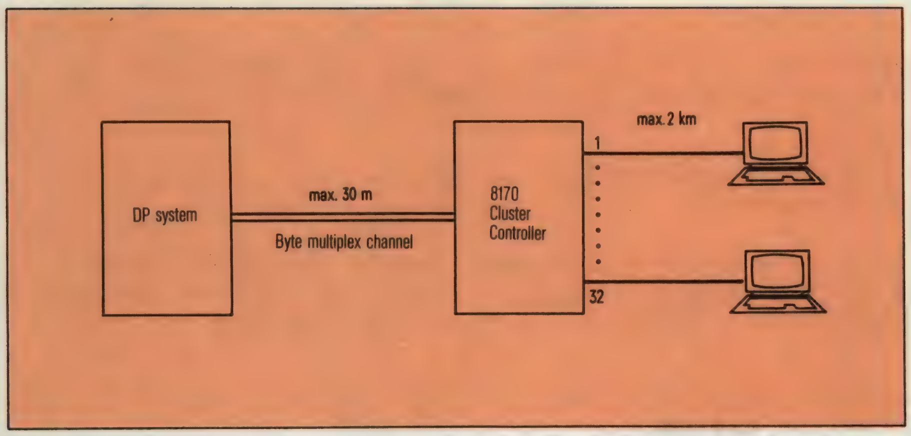


Fig. 2-3 Connection to a local cluster controller (MSN 8170)

8171 Cluster Controller

The 8171 Cluster Controller (remote) is employed far away from the host computer and in the basic version is connected via <u>one</u> remote line to the latter. By means of subsequent modification a <u>second</u> line to the host computer is possible.
Free-standing modems and integrated baseband modems are used as data transmission facilities.

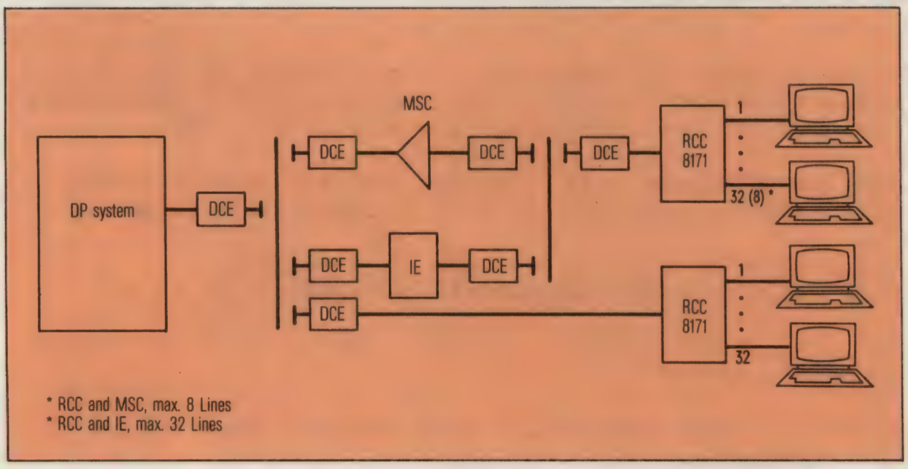


Fig. 2-4 Connection to a remote cluster controller (MSF 8171)

Course of Message Transmission

The dp system

- selects the cluster controller (local, remote) and the channel to which the desired display terminal is connected by means of addressing and

- establishes the connection to the cluster controller and gives an address to the desired channel.

Messages sent to the host computer are distinguished by two ID characters (ID1, ID2) assigned to each display terminal. In this way the host computer recognizes from which terminal messages are dispatched (see chapter 3 for further details).

2.5.2 Connection to Interface Expander

The interface expander (IE) provides a convenient and economical node for data networks. It enables a number of data display terminals to be connected to a single dp system.

The interface expander possesses

- 1 connection to the host computer
- 5 connections to display terminals.

The number of terminals which can be connected can be increased by joining two or more interface expanders, either directly or via data transmission equipment, in series.
The resulting number of terminals can be calculated according to the relationship 4n + 1 where n= the number of interface expanders.
The connection of the terminals and the host computer to the interface expanders can be achieved via

- data transmission facilities (e.g. free standing modem, integrated baseband modem) or
- via direct lines.

The generation, control and supervision of data transmission is handled by a data transmission protocol.
Only communication protocols with initiative at the host can be used in conjunction with interface expanders (e.g. polling or selecting). The display terminals connected to the interface expander are selected by means of addressing.

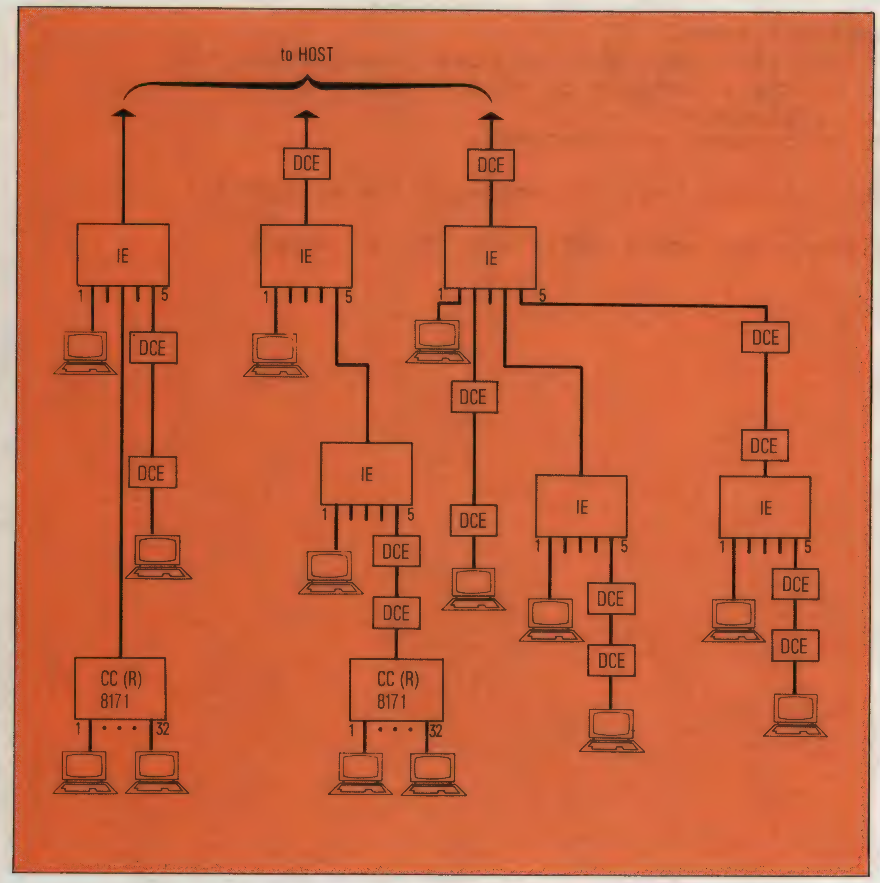


Fig. 2-5 Connection to interface expander

2.5.3 Connection to Concentrator

The concentrator enables the construction of data networks in the medium-speed transmission range.
Concentrators are particularly economical because they reduce the necessity for transmission lines and, proportionally, the data transmission facilities.

The concentrator possesses

- 1 connection to the host computer
- 60 connections to display terminals.

Extensive networks can be designed by connecting the concentrators in series.
Two concentrators KMS II can be connected to the KMS I at max. two-stage concentration.
Data interchange is then possible between maximum 178 display terminals and one host computer.
The connection of the terminals and the host computer to the concentrator can be achieved

- via data transmission facilities (free standing modem, integrated baseband modem) or
- via direct lines.

Cross-connection is also possible here.
Message interchange between the terminals and the host computer via the concentrator is safeguarded by the adherence to established regulations, namely the data transmission protocols.
The following protocols are at present provided for:

- Dialog transmission protocol (with the line programs PIN or CMUCI),
- Dialog and batch transmission procedure (MSV1/KMS or LSV1/KMS).

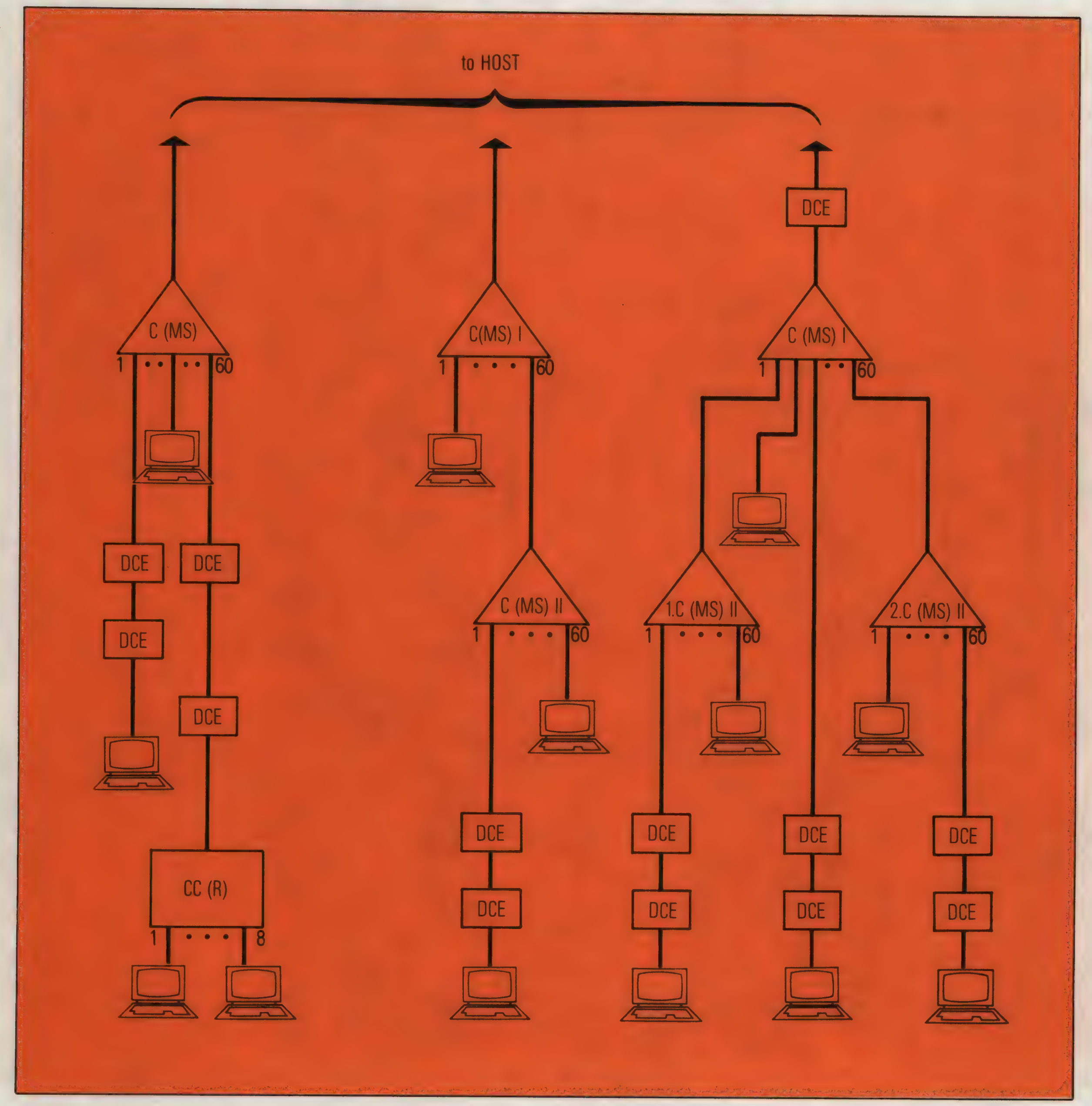


Fig. 2-6 Connection to concentrator

2.5.4 Connection to Terminal Computers

Data display terminals can be connected to terminal computers belonging to the Siemens 9.600 System

- directly

or via

- Data transmission equipment,
- 8171 Cluster Controller (remote),
- Interface Expander (IE),
- Concentrator.

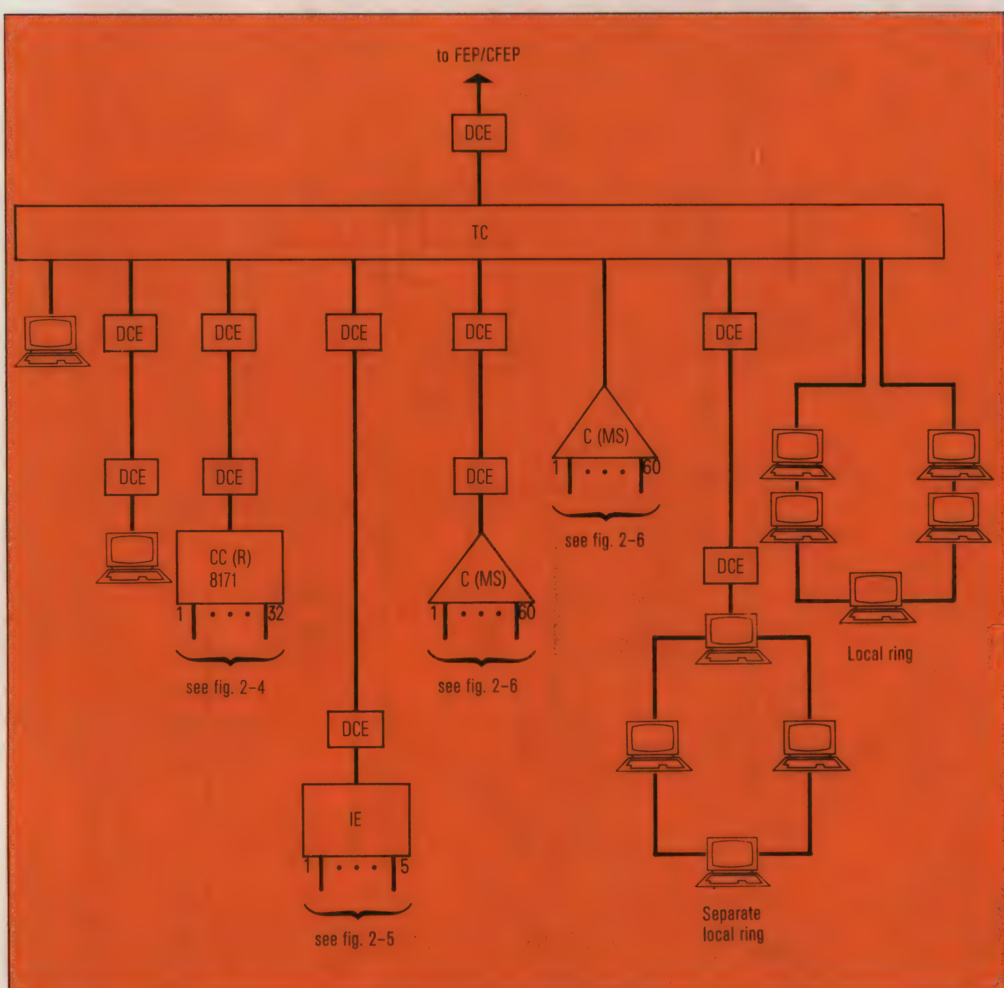


Fig. 2-7 Connection to terminal computers

2.5.5 Connection to Host Computers

Data display terminals can be connected via front-end processors and cluster controllers to System 7.500/7.700 host computers.
Display terminals can be connected directly to System 7.500 host computers via the 'Integrated Terminal Controller' (ITC), which is equivalent to a 8170 Cluster Controller (local). Moreover, an Integrated Front-End Processor (IFEP) can assume the function of the front-end processor.

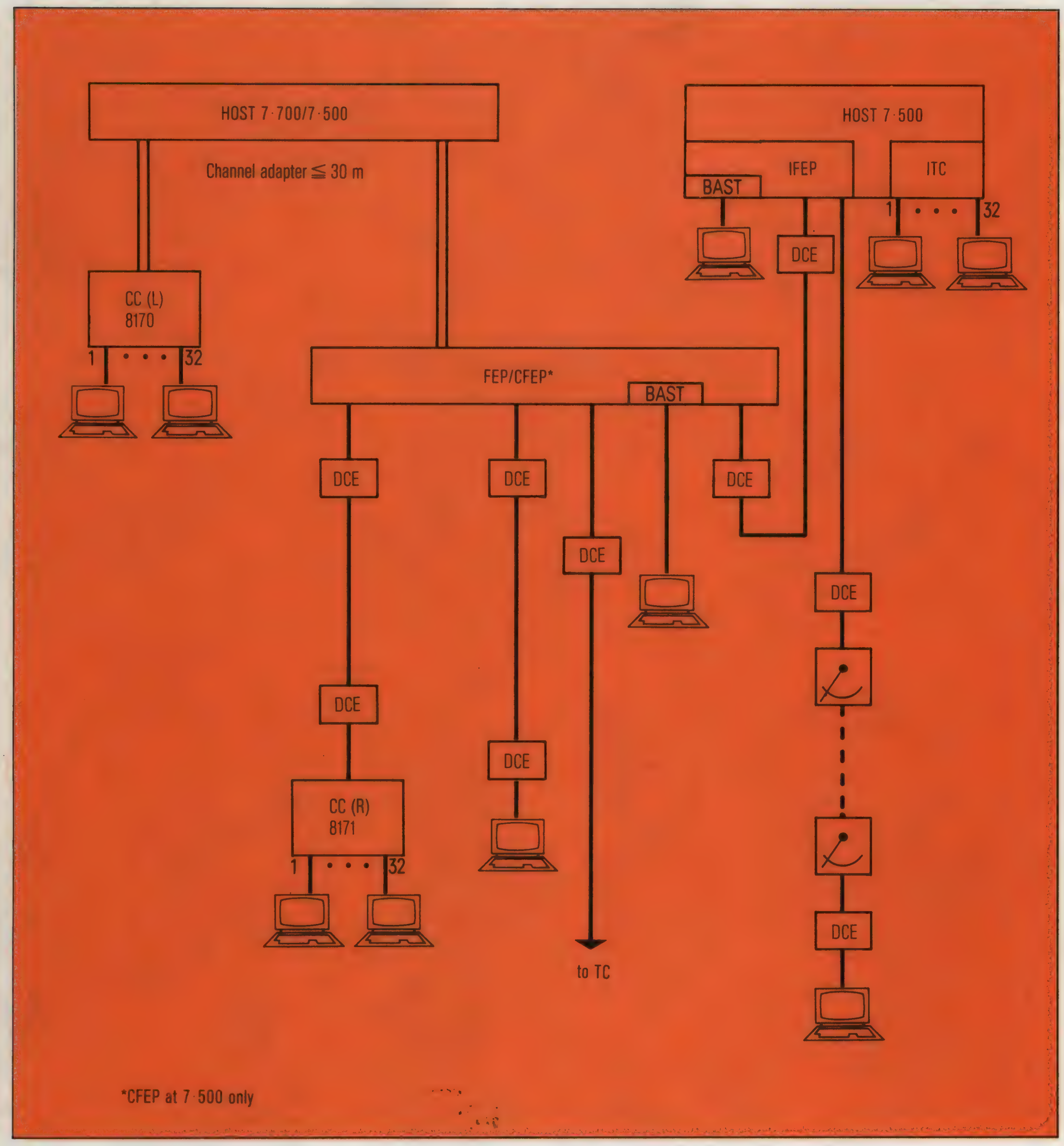


Fig. 2-8 Connection to host computers

3 MESSAGE TRANSMISSION

Transmission protocols have to be established before messages can be sent.

3.1 STAND-ALONE DATA DISPLAY TERMINAL

In the case of stand-alone terminals, message transmission is handled by the following procedure:

- Synchronous protocol MSV1

Before receiving a message a system component must be activated by specifying control characters and addresses. The host computer has here the status of a control station and only it can initiate a data transmission by sending out

- selecting sequences (host-to-terminal transmission)

or

- polling sequences (terminal-to-host transmission)

distinguished by Bit 1 of AD1/AD2.

3.1.1 Data Transmission (Host-to-Terminal)

When the host computer wishes to output a message on a display terminal it has to send a selecting sequence in advance.
The selecting sequence contains the address (AD1, AD2) of the terminal in question.
After positive acknowledgement DLE0 has been received from the display terminal, the host computer can begin sending the transmission blocks.

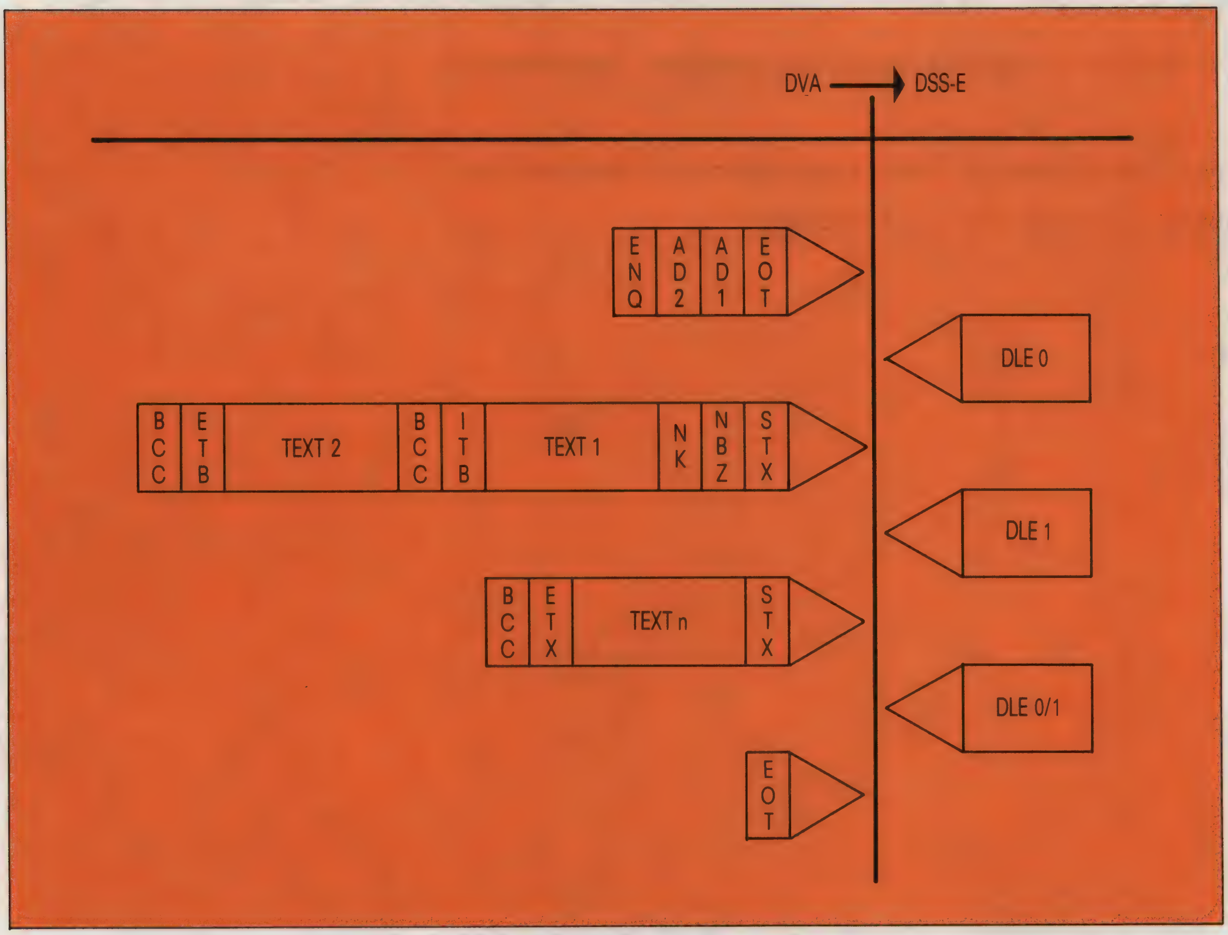


Fig. 3-1 Data transmission (selecting) DSS-E to MSF 8171

3.1.2 Data Transmission (Terminal-to-Host)

The host computer sends round a polling sequence to all the connected display terminals. The sequence contains the current address (AD1, AD2) of the requested terminal.
Should the terminal have a message ready to send, it can send it immediately. The messages are identified by ID characters (ID1, ID2).

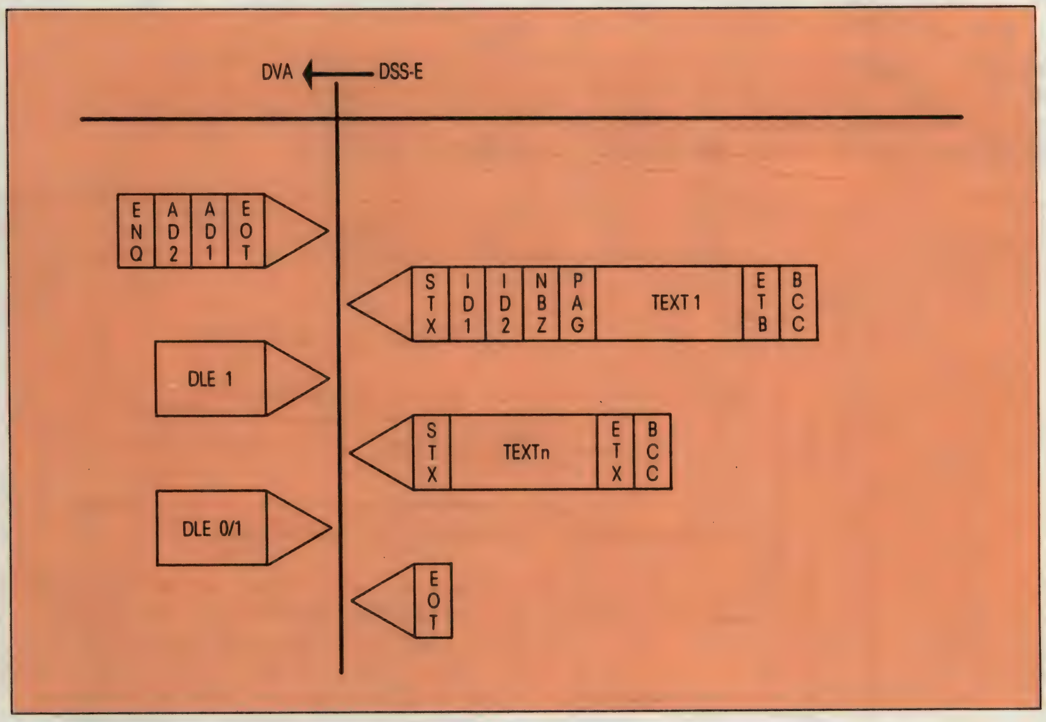


Fig. 3-2 Data transmission (polling) DSS-E to MSF 8171

3.1.3 Address and ID Formats for Standalone Terminals

The tables contain address and ID characters for addressing and identifying a standalone terminal.

The character codes are represented in hexadecimal form and when necessary as individual bits.

Operation with One Address Character

If only one address character is necessary for addressing, then the operating mode has to be activated in the presetting memory (7) via the keyboard. One has to take into consideration that this is not possible for all versions of the display terminal (see Table 2-1).

Character	Meaning	Coding	Explanation
AD1	Address 1	Bit 7 6 5 4 3 2 1	=0: Polling sequence =1: Selecting sequence
		1	Extension of station address
		0 0	Message to terminal
		0 1	Operability-checking of local printer with device address 1
		1 0	Operability-checking of local printer with device address 2
		1 1	Operability-checking of local printer with device address 3

Table 3-1 Operation with one address character

Operation with Two Address Characters

To ensure uniform addressing within the TRANSDATA 810 Terminal System, two address characters should be used.
For operation in conjunction with interface expanders, two address characters <u>must</u> be used.

Character	Meaning	Coding	Explanation
AD1	Address 1	X'40' - X'7F'	Address of standalone terminal
AD2	Address 2	Bit 7 6 5 4 3 2 1	
		1	=0: Polling sequence
		1	=1: Selecting sequence
		4 3 2	Extension of station address
		7: 1	
		6 5: 0 0	Message to terminal
		6 5: 0 1	Operability-checking of local printer with device address 1
		6 5: 1 0	Operability-checking of local printer with device address 2
		6 5: 1 1	Operability-checking of local printer with device address 3

Table 3-2 Operation with two address characters

ID Characters

Text to be sent to the host computer is given a distinguishing tag by the display terminal so that the host computer knows from which display terminal the text is coming.
The tag comprises two ID characters (ID1, ID2) that are assigned to the particular display terminal.

Character	Meaning	Coding	Explanation
ID1	Identification 1	X'40' - X'7F'	Coding as per convention with host system
ID2	Identification 2	X'40' - X'7F'	

Table 3-3 ID characters

Operation with Concentrator

Poll format: ADR

Character	Meaning	Coding	Explanation
ADR	Address character, polling	X'70'	——

Table 3-4 Poll format

Selecting format: AD1, AD2, ADE

Character	Meaning	Coding	Explanation
AD1	Address 1 AD1 + AD2 (in concentrator) specify terminal port	Bit 7 6 5 4 3 2 1 1 0 0 0 1 . . . . 0 1 1 0	=1: Selecting sequence Coding of tens position (1X - 6X) for addressing terminal ports 01 - 60 (X'43' - X'6D') =0 Single-stage concentrator, =0 (KMSI only) =1 Two-stage concentrator, =1 address = 1st KMSII =1 Two-stage concentrator, =0 address = 2nd KMSII
AD2	Address 2	7 6 5 4 3 2 1 1 0 0 0 0 1 . . . . 0 1 0 1 0	Coding of ones position (X1 - XA) for addressing terminal ports 01 - 60 (X'41' - X'6A')
ADE	Address Character, selecting	X'41'	Output to terminal
		X'51'	Operability-checking of local printer with device address 1
		X'61'	Operability-checking of local printer with device address 2
		X'71'	Operability-checking of local printer with device address 3

Table 3-5 Selecting format

3.2 CLUSTER DATA DISPLAY TERMINAL (DSS-M)

In the case clustered data display terminals, message transmission takes place in accordance with the following protocols:

- between the data display terminals and the (8170, 8171) cluster controllers, the BAM protocol (bit-serial communication protocol for connection to cluster controllers)
- between the cluster controllers and the host system, the MSV1 protocol, or for local hookups, via the software interface NEA1.

3.2.1 Message Transmission between Host Computer and Cluster Controller

<u>8170 Cluster Controller (local)</u>

The cluster controller is connected directly to the byte multiplex channel of a host computer via a standard interface cable (max. length 30 m). Data transmission is in accordance with the protocol of the dp system standard interface. The following functions, i.a., are performed:

- punctual signalling in conjunction with the data transfer on the transmission lines during byte multiplex operation,
- interpretation of host commands in the cluster controller,
- generation and interpretation of status indicators,
- supervision of character parity in the dp system.

<u>8171 Cluster Controller (remote)</u>

Data transmission via one (or two) remote lines is handled by the protected MSV1 protocol.
The host computer has control terminal status and it, alone, can initiate data transmission by sending out

- polling sequences (terminal-to-host transmission)
- selecting sequences (host-to-terminal transmission)

(distinguished by bit 1 of AD1/AD2).

3.2.2 Message Transmission between Cluster Controller and Clustered Terminal

Message transmission between the cluster controller and the clustered terminal conforms to a special intra-system protocol (BAM protocol). The controller has the initiative.
Directed by controller commands, the terminal is in one of the three protocol statuses:

- switched-on status,
- polling status,
- selecting status.

How a terminal behaves depends on its protocol status. The controller knows, by means of acknowledgments dispatched by the display terminal, if the latter has the right status and has correctly executed the commands it has received.

3.2.3 Data Transmission (Host-to-Terminal)

The controller sends an output command to the data display terminal causing the latter to adopt the selecting status and return an acknowledgment. The controller then sends the host message to the data display terminal. Each message character is acknowledged by the terminal.

3.2.4 Data Transmission (Terminal-to-Host)

The controller sends a polling command to each of the connected data display terminals in turn. If one of the terminals has been caused to adopt the 'data transmission status' (e.g. by means of the transmit key 'DÜ1'), it answers the next poll with a modified report byte.
The controller then sends a read command to this terminal and the latter adopts the polling status and acknowledges with a report byte.
With further commands the controller then requests the message characters individually from the terminal, each command being answered with one message character. The message characters received are sent straight to to the host computer by the cluster controller.

3.2.5 Addresses and IDs, 8170 Cluster Controller (Local)

Character	Meaning	Coding	Explanation
AD1 (8 bits)	Address 1	X'E0' - X'FF'	Channel address of terminal
AD2 (8 bits)	Address 2	X'CF'	Message to terminal
		X'DF'	Operability-checking of local printer with device address 1
		X'EF'	Operability-checking of local printer with device address 2
		X'FF'	Operability-checking of local printer with device address 3
			The channel address of the 8170 on the byte multiplex channel must be specified in the parameter XLINE (LINE=) in the CCP for 8170 operation. The MUX channel address must be set on pc board SA3-K in the 8170 (= line number in communication system).
ID1	Identification 1	ID1=AD1	ID1, ID2 and NKE1/2 are not sent to the communication user program if transmission is from terminal to host. ID1 and ID2 are generated by the 8170.
ID2	Identification 2	X'CF'	

Table 3-6 Addresses and IDs, 8170 Cluster Controller

Coding for Channel Address AD1 of 8170 Cluster Controller

Line connector	1				2				3				4				5				6				7				8			
Channel address AD1	E0	E2	E4	E6	E8	EA	EC	EE	F0	F2	F4	F6	F8	FA	FC	FE	E1	E3	E5	E7	E9	EB	ED	EF	F1	F3	F5	F7	F9	FB	FD	FF
Terminal no.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32

Table 3-7 Coding for channel address AD1 of the 8170 Cluster Controller

3.2.6 Addresses and IDs, 8171 Cluster Controller (Remote)

Character	Meaning	Coding	Explanation
AD1	Address 1	X'40'	General poll. 8171 (polling only)
		X'60' - X'7F'	Specific poll, terminal (channel address of 8171-controlled terminal)
AD2	Address 2	Bit 7 6 5 4 3 2 1	=0: Polling sequence (SA) =1: Selecting sequence (EA)
		1	Station address of 8171 in a multipoint circuit
		0 0	Message to terminal
		0 1	Operability-checking of local printer with device address 1
		1 0	Operability-checking of local printer with device address 2
		1 1	Operability-checking of local printer with device address 3
ID1	Identification 1	X'60' - X'7F'	Channel address of 9171-controlled terminal, generated in 8171
ID2	Identification 2	X'40' - X'7F'	ID of 8171 Cluster Controller (remote): - In the case of general polling of the 8171, ID character interpretation must be performed by the host. - In the case of specific polling of a terminal via the 8171, interpretation is optional.

Table 3-8 Addresses and IDs, 8171 Cluster Controller

Coding for Channel Address AD1 of the 8171 Cluster Controller

Line connector	1				2				3				4				5				6				7				8			
Channel 1) address AD1	40	42	44	46	48	4A	4C	4E	50	52	54	56	58	5A	5C	5E	41	43	45	47	49	4B	4D	4F	51	53	55	57	59	5B	5D	5F
Channel 2) address AD1	60	62	64	66	68	6A	6C	6E	70	72	74	76	78	7A	7C	7E	61	63	65	67	69	6B	6D	6F	71	73	75	77	79	7B	7D	7F
DSS-No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32

1) Bit 6 = 0: General polling of the 8171 Cluster Controller (polling only)

2) Bit 6 = 1: Specific polling of the 8171 Cluster Controller (polling and selecting)

Table 3-9 Coding for channel address AD1 of the 8171 Cluster Controller

4 MESSAGE FORMAT

A message consists of the following subsections:

- message header and
- text blocks.

The following table gives an overview of structure and formatting options.

Abbreviations used in the table:
SOM = start-of-message character MT = message text
MH = message header PAR = parameter entries

Message header		Text block	Remarks
SOM	-	MT	e.g. simple dialog
SOM	MH	MT	e.g. parameter loading
SOM	-	nPAR	
SOM	-	nPAR + MT	e.g. message with special commands
SOM	-	MT + nPAR	
SOM	-	nPAR + MT + nPAR	
SOM	-	MT + nPAR + MT	

Table 4-1 Message format

The message text (MT) can be separated by means of

- absolute positioning commands (IS4 sequences),
- field separators (IS2/IS3 sequences),
- device functions (ESC sequences).

<u>Note</u>
All the codes in this chapter - in hexadecimal notation (X'') and single-bit form - comply with the ISO 7-bit code, for which the abbreviation 'ISO-7' is used in the tables.

4.1 MESSAGE PREFIX

A message prefix consisting of at least the start-of-message character (SOM) is present in <u>every</u> I/O operation.

4.1.1 Message Prefix (Host-to-Terminal Transmission)

In the case of transmission from host-to-terminal, the message prefix comprises at least the start-of-message character (SOM).
The SOM specifies the length of the subsequent message header.
If the SOM = X'40', the SOM is not followed by a message header.
The length of parameter entries (PAG) is not specified in the SOM.

Direction of transmission	SOM	Meaning
Host-to-terminal	X'40'	No message header
	X'48'	Message header = PARAM0
	X'50'	Message header = PARAM0 + PARAM1
	X'58'	Message header = PARAM0 + PARAM1 + PARAM2

Table 4-2 Message prefix (host-to-terminal transmission)

Message Header (Host-to-Terminal Transmission)

The message header is part of the message prefix and may be up to 24 bytes long (3 x 8 bytes). The parameter ranges PARAM0, PARAM1 and PARAM2 are loaded by means of the message header. These ranges have a format equivalent to the parameter ranges PAR 00L, PAR 10L and PAR 20L.

During host-to-terminal transmission, entries of PARAM0, PARAM1 and PARAM2 are entered in the

- parameter ranges for functions, initiated by the terminal user via the keys (PAR 00L, PAR 10L and PAR 20L) and

- parameter ranges for host-initiated functions (PAR 00D, PAR 10D, PAR 20D).

When the message header is used, the parameter ranges - in contrast to the application of PAG parameter entries - cannot be loaded selectively.
If a new message header is output, the contents of all previously loaded parameters are erased and replaced by the contents of the new message header.

Format and Functions of Parameter Ranges

PARAM0: Format corresponds to that of PAR 00L/00D.
PARAM0 is used exclusively for communication between the data display terminal and the host and contains device-specific information on how the subsequent text is to be handled.

PARAM1 and PARAM2: Format of these ranges correspond to those of PAR 10L/10D and PAR 20L/20D.
PARAM1 and PARAM2 are identical in format and contain information on communication between the data display terminal and the local printer or printer terminals connected to the cluster controller.
Control information can be sent directly from the host to the printer terminals via PARAM1.

Possible Error Sources

- If a message header is output, the contents of <u>all</u> parameter ranges loaded via parameter entries are cleared and replaced by the contents of the message header.

- If an intra-system data exchange is initiated with a message header in PAR 00L (=PARAM0), in character 4 (GEF1), by entering the function LA1, then an acknowledgment in PAR 10L (=PARAM1) is <u>not</u> permitted. The request for acknowledgment has to be entered in PAR 10D, in character 2 (PBH).

- When using a message header, only two-character sequences may be entered in character 1 of PAR 00D. In the case of terminal-to-host transmission, the queue control character WAR (character 8 of PAR 00L) is always in the ISO 7-bit code, i.e. the bit combinations X'7F' are falsified. Bit 8 is set to 0.

4.1.2 Message Prefix (Terminal-to-Host Transmission) Komt in ISO terug.

In the case of terminal-to-host transmission, the message prefix consists of, at least, the SOM character and an 8-byte long message header (NK=PAR 00E).
If a text ID is also dispatched, the message header is 16 bytes long (NK=PAR 00E + PAR 01E ~~+ PAR 02E~~).
In the case of field mode (e.g. BERMUDA), the message header is 24 bytes long.
(NK=PAR 00E + PAR 01E + PAR 02E).

Direction of transmission	SOM	Meaning
Terminal-to-host	X'48'	Header comprises PAR 00E
	X'50'	Header comprises PAR 00E + PAR 01E
	X'58'	Header comprises PAR 00E + PAR 01E + PAR 02E

Table 4-3 Message prefix (terminal-to-host transmission)

Message Header (Terminal-to-Host Transmission)

The message header is part of the message prefix and may be up to 16 bytes or, in the case of field mode, 24 bytes long.
The following parameter ranges are loaded by means of this header:

- PAR 00E,
- PAR 01E (for operation with text ID),
- PAR 02E (for field mode; character 6 (FST4) of PAR 00L=X'42').

For each transmission to the host, the display terminal places these parameter ranges before the message, depending on character 5 (FST3) of PAR 00L.

Functions of Parameter Ranges

PAR 00E: Device-specific information is entered in this range.

PAR 01E: This parameter range is the extension of PAR 00E and is for entering the header length, specifying the cursor ~~length~~ before transmission is initiated and for noting error messages. position

PAR 02E: This parameter range is reserved for field mode.

Format of Parameter Range PAR 00E

Kand in ISO terry

Character	Designation	Meaning
1	SAW	Send command
2	PFLNG	Buffer length
3	ZLNG	Line length
4	AUSG	Program revision level
5	ZZ1	Status character 1
6	ZZ2	Status character 2
7	CDS	Send-key code
8	WAR	Queue control character

Table 4-4 Overview of the characters of PAR 00E

Char.	Design.	Meaning	Coding	Explanation
1	SAW	Send command	Dependent on whether SAW1 or SAW2 is used.	Depending on the transmit key used (DÜ1 or DÜ2), the corresponding send command SAW1 or SAW2 is entered in PAR 00E and executed. The functions SAW1 and SAW2 are specified (via host-to-terminal transmission) in PAR 00L and PAR 01L respectively. After terminal switchon or user initiation of the 'LSP' function, SAW=X'00'.
2	PFLNG	Buffer length	X'40'	——
3	ZLNG	Line length	X'30'	——
4	AUSG	Program revision level	X'60'	Program memory: DIALOG Program memory: DIALOG + BERMUDA
			X'67'	Program memory: DIALOG
			X'64'	Program memory: DIALOG
			X'65'	Program memory: DIALOG + BERMUDA with key verification.
			X'66'	Program memory: DIALOG for X.21
5	ZZ1	Status char. 1. Contains user ID (Badge reader and keylock switch) and is an extension of ZZ2.	Bit 7 6 5 4 3 2 1 Bit 7: 1	
			Bit 1	=0: no badge reader ID present =1: badge reader ID present (badge inserted) 1)
			Bit 2	=0: / =1: Keylock switch 1 2)
			Bit 3	=0: / =1: Keylock switch 2 2)
			Bit 4	=0: / =1: Keylock switch 3 2)
			Bit 5	=0: / =1: Local device with GAD6 3)
			Bit 6	=0: / =1: Local device with GAD7 3)

Char.	Design.	Meaning	Coding	Explanation	
6	ZZ2	Status char. 2.	Bit 7 6 5 4 3 2 1 Bit 7: 1 Bit 1	=0: =1: Keyboard	3)
			Bit 2	=0: =1: Local device with GAD1	3)
			Bit 3	=0: =1: Local device with GAD2	3)
			Bit 4	=0: =1: Local device with GAD3	3)
			Bit 5	=0: =1: Local device with GAD4	3)
			Bit 6	=0: =1: Local device with GAD5	3)
7	CDS	Send-key code. Code of actuated key entered in char.	See Table 4-33 "Send functions".	Transmission as per Table 4-33 "Send functions".	
8	WAR	Queue control character	Optional but no transmit control characters.	The terminal returns the host-specified WAR, unmodified, to the host.	

1) Bit 1 is reset by a K14 message

2) 0: Keylock switch 'OFF'
   1: Keylock switch 'ON'

3) 0: Device inoperable, i.e.
   - disconnected or
   - malfunctioning (device-specific status message)
   1: Device operable

Table 4-5 Description of the characters in the parameter range PAR 00E

Format of Parameter Range PAR 01E Kunt in ISO terug

Char.	Design.	Meaning	Coding	Explanation
1	TIL	Text ID length ZOALS IN TIL VAN PAR 01L	X'40'	No text identification TI
			X'41'	TI = 8 bytes (PAR 01D)
			X'42'	TI = 16 bytes (PAR 01D + PAR 02D)
			X'43'	TI = 24 bytes (PAR 01D + PAR 02D + PAR 03D)
2	TKL	Header length	X'00'	Reserved
3	ZLA	Line address	Dependent on FST3 of PAR 00L (bit 2)	Characters 3, 4 and 5 indicate cursor position before activation of a DÜ, K or F key.
4	SPA	Column address		
5	SAD	Page address	X'30'	
6	FEM	Error message	See Table 4-7	For error messages
7	——	——	X'00'	Reserved
8	——	——	X'00'	Reserved

Table 4-6 Character description of PAR 01E

Text Identification

Text ID (TI) can be structured freely by the programmer. In this way, transmissions (e.g. terminal-to-host) from different display terminals can receive different text IDs and be assigned in a particular manner.
In the case of host-to-terminal transmission, a text ID will be stored in the parameter portions PAR 01D, PAR 02D and PAR 03D (8 bytes each).
The text ID length (TIL) indicates whether a text ID shall be sent to the host from the terminal parameter portions PAR 01D - PAR 03D and the length of the ID.
If a text ID is stored in the parameter portions, it is sent to the host - every time transmission to the host occurs - immediately after PAR 01E, depending on the state of character 1 (TIL) of PAR 01L.

Error Messages

If an errored message or an invalid function is sent by the host, appropriate error messages are recorded in character 6 (FEM) of PAR 01E by the terminal.
The terminal hands over this parameter range when it is requested during the next transmission to the system.

Coding	Function	Remarks
X'40'	MAR	Non-markable field
X'41'	WDH	Invalid address
X'42'	WDH	Non-null characters preceding logical end of line
X'44'	LVA	Invalid address
X'46'	PAR	Range full
X'47'	PAR	Invalid characters
X'48'	PAR	Invalid function
X'4A'	P	P range full (16 characters max.)
X'4B'	P	Invalid character
X'4C'	P	Invalid function
X'4F'	PAR OOD	Printer control character (ESC seq.) in PAR OOD
X'50'	PAK	More than 48 display control chars. in system line
X'51'	-	ESC sequence invalid or incomplete
X'52'	PAK	Parameter entries invalid or incorrect
X'53'	FTZ	IS2 or IS3 sequence invalid at this location
X'54'	Position	Invalid positioning information
X'55'	DÜ	Invalid function in message
X'56'	PAR OOD	Invalid function in PAR OOD
X'57'	LA	Invalid function in message
X'5A'	SS	Invalid function in PAR OOD
X'60'	Host-to-terminal transmission	Characters preceding logical end of line
X'61'	EFG	Character insertion in protected field
X'62'	AFG	Character deletion in protected field or after logical end of line
X'63'	LZF	Invalid character in protected field
X'64'	LVD	Invalid character in protected field as far as to end of screen
X'65'	EFG	Alphabetic character insert in protected field
X'66'	RUB	Roll-up function blocked in RUB mode
X'67'	Host-to-terminal transmission	Key verification illegal in protected field
X'6A'	TES	Invalid entry
X'6B'	AUS/RUF	X.21 functions from host invalid
X'71'	LA	Incorrect printer terminal parameters
X'75'	LA	Incorrect command in parameter range
X'79'	LA	No text in bypass mode
X'7E'	Hardware	General hardware error
X'7F'	RM	Acknowledgment buffer overflow

Table 4-7 Error messages

4.1.3 Message Prefix when using Acknowledgment Characters

When acknowledgment characters are used, the message is composed only of a 4-byte-long message prefix.

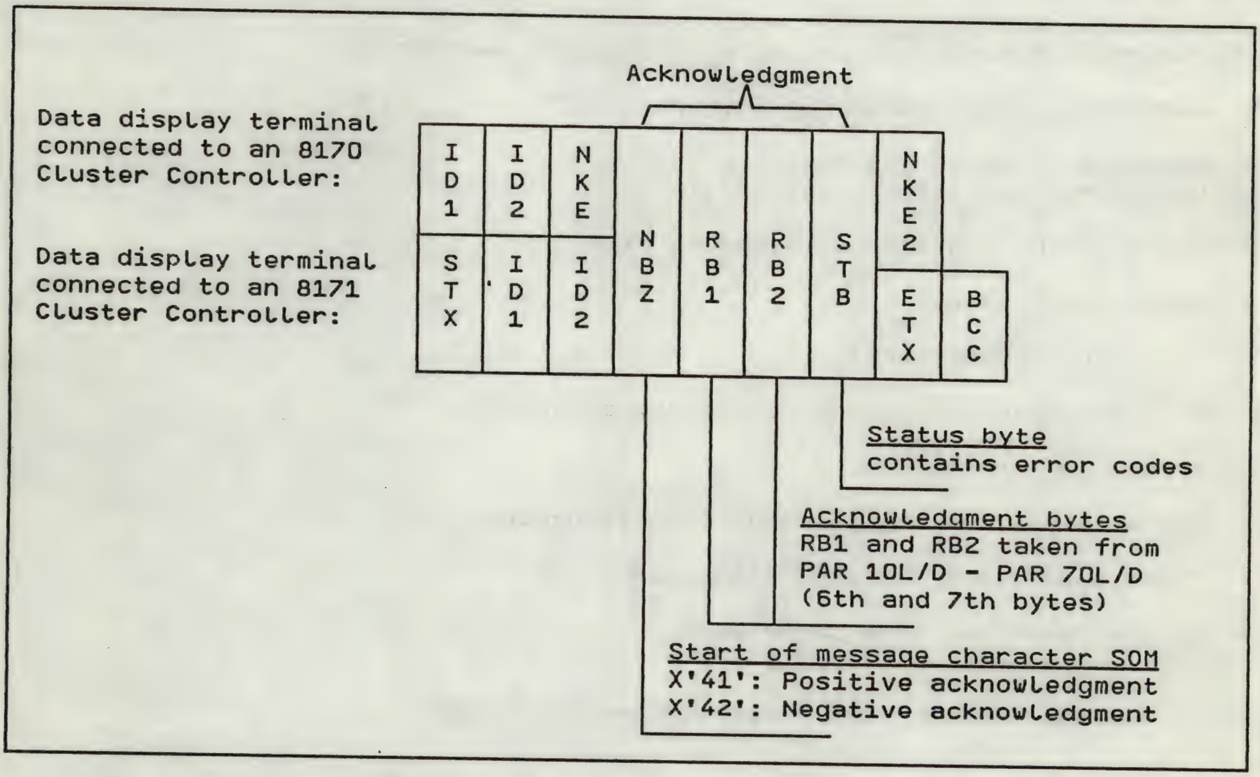


Fig. 4-1 Acknowledgment from local devices

4.2 MESSAGE SECTION

Message sections can consist of

- parameter entries without messages
  (loading of parameter ranges and parameter sections),
- messages without parameter entries,
- parameter entries and messages
  (messages with special commands).

Messages can be constructed by means of

- text, comprising
  - numeric characters,
  - alphabetic characters (large and small),
  - special characters,
- absolute positioning commands (IS4 sequences),
- field separators (IS2/IS3 sequences),
- device functions (ESC sequences).

4.2.1 Message Section with Parameter Entries (PAG)

The parameter entries are no longer part of the message header.
The length of parameter entries are not noted in the start-of-message character (SOM); the SOM = X'40'.
Parameter ranges and parameter sections can be exactly loaded.
Parameter ranges and parameter sections are specific memory areas within the terminal by which the operational behaviour of the terminal is controlled.
The following can be entered by means of parameter entries:

- device functions
- send commands
- text IDs
- operational statuses.

Once loaded, parameter ranges remain valid until they are overwritten by new parameter entries or by a message header or, depending on the status of character 3 of PAR OOL, until they are erased via the 'LSP' function.
When the display terminal is switched off, the parameter range contents are erased whereas at switchon, default values are automatically loaded within the display terminal.

Format of Parameter Entries

Parameter entries are identified by a parameter identifier character (PAK). The subsequent address entries of

- parameter range and
- parameter section

control the exact storage of the parameter entries in the appropriate parameter ranges and parameter sections within the display terminal.

<— Direction of transmission (host-to-terminal)

SOM	ESC	Space	PAK	Address of para-meter range	Address of para-meter section	Para-meter entries 8 bytes	Para-meter entries 8 bytes	Para-meter entries 8 bytes	Para-meter entries 8 bytes	End character	Text or further para-meters

X'21'

3rd extension
PAR 03D

2nd extension
PAR 02L/02D

1st extension
PAR 01L/01D

Base parameter
PAR00L/00D - PAR70L/70D

X'40': Base parameter
X'41': 1st extension
X'42': 2nd extension
X'43': 3rd extension

X'40' - X'47': Parameter ranges for user-initiated functions
X'48' - X'4F': Parameter ranges for host-initiated functions after transmission close

Parameter identifier character EBCDIC = 81 ASCII = 61

Identifier character for a three-character sequence SPACE

Code extension character ESC

Start-of-message character;
always X'40' when using PAG (no message header)

Fig. 4-2 Format of a message including parameter entries

If a number of parameter ranges are filled successively, each range must be identified by the parameter identifier (ESC, space, PAK) and terminated with the end character (!). This also applies when text portions occur between the parameter entries.

Functions of Parameter Ranges and Parameter Sections

The following tables give an overview of the functions of parameter ranges and sections.

Parameter range	Function
PAR 00L	The parameter range 00L contains values for text manipulation and function control. In the case of terminal-to-host transmission, character FST3 of PAR 00L specifies the length of the message header.
PAR 00D	Used for device functions that are to be executed after transmission has been completed. Host-initiation only. Character LAP1 may be used for setting bypass mode. Character AZL may be used for activating the system status line.
PAR 10L through PAR 70L	Used for entering a print command or the channel address of a printer terminal connected to the same cluster controller. Initiation is via the key corresponding to the parameter range. Example: Activation of key LA2 initiates the function entered in parameter PAR 20L.
PAR 10D through PAR 70D	Format as for PAR 10L through PAR 70L. LA key function initiated by the host.

Table 4-8 Parameter range functions

Parameter section	Function
PAR 01L	This range contains values for text manipulation and function control.
PAR 02L	Used for field-related specifications in field mode only.
PAR 01D through PAR 03D	Used for text IDs that are included in terminal-to-host transmission. The length of the text ID is given in character TIL of PAR 01L.

Table 4-9 Parameter section functions

Coding of Parameter Ranges and Sections

Abbreviations used in the table:

PR = Parameter range
PS = Parameter section

Parameter ranges for functions that can be initiated by the terminal user via keys (hexadecimal coding)

Key	GEF	LA1	LA2	LA3	LA4	LA5	LA6	LA7
	↓	↓	↓	↓	↓	↓	↓	↓
PR / PS	40	41	42	43	44	45	46	47
X'40'	PAR 00L	PAR 10L	PAR 20L	PAR 30L	PAR 40L	PAR 50L	PAR 60L	PAR 70L
X'41'	PAR 01L	-	-	-	-	-	-	-
X'42'	PAR 02L	-	-	-	-	-	-	-

x'50' zie blz 4-59

Parameter ranges for host-controlled functions that are not initiated by the system until after transmission (hexadecimal coding)

Function	GEF	LA1	LA2	LA3	LA4	LA5	LA6	LA7
	↓	↓	↓	↓	↓	↓	↓	↓
PR / PS	48	49	4A	4B	4C	4D	4E	4F
X'40'	PAR 00D	PAR 10D	PAR 20D	PAR 30D	PAR 40D	PAR 50D	PAR 60D	PAR 70D
X'41'	PAR 01D	-	-	-	-	-	-	-
X'42'	PAR 02D	-	-	-	-	-	-	-
X'43'	PAR 03D	-	-	-	-	-	-	-

Table 4-10 Coding of parameter ranges and sections

Possible Error Sources

- One parameter range (max.) with two or three parameter sections can be loaded with <u>one</u> parameter entry sequence. (4 x 8 bytes = base parameter + 1st, 2nd and 3rd extension).
- The end character <u>must</u> be located at the end of a parameter entry sequence. If no end character is included, any characters entered after the addressed parameter range (section) are lost.
- If the parameter ranges are loaded by means of a <u>message header</u>, parameter ranges loaded via <u>parameter entries</u> are erased.

4.2.1.1 Format of Parameter Range PAR 00L

The parameter range PAR 00L can be loaded by means of parameter entries (PAG) as well as PARAMO of the message header (MH).
If a message header is used, the contents of PARAMO are moved to the terminal parameter ranges PAR 00L or PAR 00D.

Character	Designation	Meaning
1	SAW1	Send command 1
2	FST1	Function control character 1
3	FST2	Function control character 2
4	GEF1	Device function character 1
5	FST3	Function control character 3
6	FST4	Only with field mode (BERMUDA)
7	GEF2	No meaning assigned
8	WAR	Queue control character

Table 4-11 Survey of the characters of PAR 00L

Char.	Desig.	Meaning	Coding	Explanation
1	SAW1	Send command 1		The character SAW1 is interpreted when DÜ1 or F functions are initiated. In the case of transmission from host to terminal, the code of the send command used is moved to PAR 00E. Invalid codes are dealt with as 'Send unprotected fields'.
			X'40'	Send unprotected fields
			X'41'	Send modified fields
			X'42'	Send unprotected fields from cursor
			X'43'	Send display buffer contents including all IS sequences
			X'44'	Send unprotected fields, without NUL characters
			X'45'	Send modified fields, without NUL characters
			X'46'	Send unprotected fields (from cursor), without NUL characters
			X'47'	Reserved for field mode
			X'48'	Send from start marker to cursor, without NUL characters; the last start character preceding the cursor is valid.
			X'4C'	Send unprotected fields, with relevant NUL characters
			X'4D'	Send modified fields, with relevant NUL characters
			X'4E'	Send unprotected fields (from cursor), with relevant NUL characters
			X'4F'	Send only modified fields, without NUL characters

Char.	Desig.	Meaning	Coding	Explanation
2	FST1	Function control char. 1	Bit 7 6 5 4 3 2 1 v 1	Entry of functions EFZ and AFZ
			Bit 1	=0: via keyboard enabled =1: via keyboard disabled
			Bit 2	=0: Field separators and protected data are copied from last line to cursor line. Unprotected data is replaced by NUL characters. =1: Field separators and all data are copied from last line to cursor line.
			Bit 3	=0: Field separators and protected data are copied from cursor line to last line. Unprotected data is replaced by NUL characters. =1: Field separators and all data are copied from cursor line to last line.
			Bit 4	=0: Cursor function: cursor cannot be moved to protected fields. =1: Cursor function: cursor can be moved to protected fields.
			Bit 5	=0: RU (Roll-up) function enabled =1: RU (Roll-up) function disabled
			Bit 6	=0: RUB (Roll-up) mode off =1: RUB (Roll-up) mode on

Char.	Desig.	Meaning	Coding	Explanation
3	FST2	Function control char. 2	Bit 7 6 5 4 3 2 1 1 (bit 7)	LSP (Erase memory) function
			Bit 1	=0: keyboard initiation enabled =1: keyboard initiation disabled
			Bit 2	=0: parameter ranges PAR 10L - PAR 70L not erased when LSP function is initiated. =1: parameter ranges PAR 10L - PAR 70L erased when LSP function is initiated.
			Bit 3	=0: Autotab (ATAB) is off. =1: Autotab (ATAB) is on.
			Bit 4	=0: FAZ function (field separator to defined state) initiated after every output. =1: FAZ function is initiated only when FAZ is output in message.
			Bit 5	See subsequent Table 4-14
			Bit 6	=0: If LVD (Erase variable data) is entered, modify and flash bits (MOD and BLINKEN) in unprotected fields are erased between cursor and end marker. =1: If LVD is entered, modify and flash bits in unprotected fields are erased between start of screen and end of screen.
4	GEF1	Device function char. 1	See section 4.2.4	Character 2 (device function) is moved to character 1 (GEF1) of parameter range PAR 00D (only for terminal-to-host transmission via message header). *

Bij parameter entries PAG (dus achter message header) deze opnemen in PAR00D

Char.	Desig.	Meaning	Coding	Explanation
5	FST3	Function control char. 3	Bit 7 6 5 4 3 2 1 (bit 7 = 1); Bit 1	=0: Cursor position tracks during host-to-terminal transmission. =1: Cursor position controlled via device function SS (Set cursor).
			Bit 2	=0: For terminal-to-host transmission, positioning commands and cursor position are specified in PAR 01E in 8161 compatible code (Table 4-19, version B). =1: For terminal-to-host transmission, positioning commands and cursor position are specified in PAR 01E according to coding in Tab. 4-19 version A.
			Bit 3	=0: For terminal-to-host transmission, the message header comprises only PAR 00E. =1: For terminal-to-host transmission, the message header comprises PAR 00E and PAR 01E. *1
			Bit 4	=0: After terminal-to-host transmission, the cursor is behind the valid end marker or at the start of screen. =1: Cursor position is not affected by terminal-to-host transmission.
			Bit 5	=0: Dialog mode not activated. =1: Dialog mode activated. After initiation of a transmission function via keyboard, the keyboard is locked. A transmission from host that involves the display buffer unlocks the keyboard.
			Bit 6	=0: Keyboard released =1: Keyboard locked Short messages (Kx) to host always possible.

*1 Jn Tiam mode : K2 geeft geen break !

Char.	Desig.	Meaning	Coding	Explanation
6	FST4	Function control char. 4	X'40'	Dialog mode
			X'42'	Field mode (BERMUDA) only
7	GEF2	Device function char. 2	——	No meaning assigned
8	WAR *	Queue control char.	Any coding but no transmission control char.	In terminal-to-host transmission, returned unmodified to the system.

Table 4-12 Description of characters of PAR OOL

* by line mode x'41' (ASCII CODE)

anders x'40' (TIAM)

x'44' (FHS)

Send Data in Formatted Mode

In formatted mode, data is transmitted, depending on the field handling character (FBZ) and the send command (SAW), according to the following table.

<table>
<tr><td colspan="3">Code of SAW 1)</td><td>48</td><td>40</td><td>44</td><td>4C</td><td>42</td><td>46</td><td>4E</td><td>41</td><td>45</td><td>4D</td><td>4F</td><td>43</td></tr>
<tr><td colspan="3" rowspan="4"></td><td colspan="7">Send unprotected fields</td><td colspan="4">Send modified fields</td><td>Send display buffer contends</td></tr>
<tr><td>from start marker/start of screen</td><td colspan="3">from start of buffer</td><td colspan="3">from cursor</td><td colspan="4">from start of buffer</td><td>from start</td></tr>
<tr><td>to cursor</td><td colspan="10">to end marker or end of buffer</td><td>to end 3)</td></tr>
<tr><td>with relevant NULs</td><td>with NULs</td><td>with-out NULs</td><td>with rel. NULs</td><td>with NULs</td><td>with-out NULs</td><td>with rel. NULs</td><td>with NULs</td><td>with-out NULs</td><td>with rel. NULs</td><td>without NULs</td><td>with NULs</td></tr>
<tr><td colspan="3">Bit set in FBZ</td><td rowspan="2"></td><td rowspan="2"></td><td rowspan="2"></td><td rowspan="2"></td><td rowspan="2"></td><td rowspan="2"></td><td rowspan="2"></td><td rowspan="2"></td><td rowspan="2"></td><td rowspan="2"></td><td rowspan="2"></td><td rowspan="2"></td></tr>
<tr><td>1</td><td>6</td><td>3</td></tr>
<tr><td rowspan="4">0</td><td rowspan="2">0</td><td>0</td><td rowspan="4">Address of start marker or start of screen + start marker + field addresses + field contents 2)</td><td rowspan="4">Field contents</td><td colspan="2" rowspan="4">Field addresses + field contents</td><td rowspan="4">Cursor address + field contents</td><td colspan="2" rowspan="4">Cursor address + field addresses + field contents</td><td>—</td><td>—</td><td>—</td><td>—</td><td rowspan="8">Text + field separators</td></tr>
<tr><td>1</td><td colspan="3" rowspan="3">Field adresses + field contents</td><td>Field addr. + field contents</td></tr>
<tr><td rowspan="2">1</td><td>0</td><td>—</td></tr>
<tr><td>1</td><td>Field addr. + field contents</td></tr>
<tr><td rowspan="4">1</td><td rowspan="2">0</td><td>0</td><td>—</td><td>—</td><td>—</td><td>—</td><td>—</td><td>—</td><td>—</td><td>—</td><td>—</td><td>—</td><td>—</td></tr>
<tr><td>1</td><td>—</td><td>—</td><td>—</td><td>—</td><td>—</td><td>—</td><td>—</td><td colspan="3" rowspan="3">Field addresses</td><td>Field addr. + field contents</td></tr>
<tr><td rowspan="2">1</td><td>0</td><td>—</td><td>—</td><td>—</td><td>—</td><td>—</td><td>—</td><td>—</td><td>—</td></tr>
<tr><td>1</td><td>—</td><td>—</td><td>—</td><td>—</td><td>—</td><td>—</td><td>—</td><td>Field addr. + field contents</td></tr>
</table>

1) Hexadecimal coding
2) An end marker is considered a data character.
3) No restrictions

Table 4-13 Send data in formatted mode

Explanation

- EM: Only end markers in send fields are interpreted.
- Relevant null characters: Null characters are relevant null characters if they identify an interval, i.e. if at least one send character or end marker is included in the same field.
- LZE: Characters between a logical end-of-line (LZE) marker and the end of a field or the physical end of a line are not sent.

Handling of Modify and Flash Bits

When the function LZF is entered, the modify and flash bits are set, cleared or remain unmodified depending on the status of character 3 of PAR 00L (FST2).

Meaning of abbreviations in the table:

M = modify bit
B = flash bit
↓ = cleared
↑ = set
- = unchanged

Handling of modify and flash bits		FST2			
		Bit 5 = 0		Bit 5 = 1	
from	to	M	B	M	B
Start of field	End of field	↓	↓	↑	—
Start of field	Any position in field	↓	↓	↑	—
Any position in field	Any position in field	—	—	↑	—
Any position in field	End of field	—	—	↑	—

Table 4-14 Handling of modify and flash bits

4.2.1.2 Format of Parameter Range PAR 01L

The parameter range PAR 01L is used as an extension to the base parameter range PAR 00L and is loadable only by means of parameter entries.

Character	Designation	Meaning
1	TIL	Text ID Length
2	TKL	Reserved
3	FST5	Causes LA keys to be blocked
4	FST6	Causes P keys to be blocked
5	FST7	Programming capability for P keys disabled
6	SAW2	Send command SAW2
7	FST8	Request for badge information entry
8	—	Reserved

Table 4-15 Survey of the characters of PAR 01L

Char.	Desig.	Meaning	Coding	Explanation
1	TIL	Text ID length indicates whether and to what extent a TI is to be sent from the mailbox facilities (BK) to host during next transm. to host (BK = PAR 01D to PAR 03D).	X'00'	No text ID
			X'41'	TI = 8 bytes (PAR 01D)
			X'42'	TI = 16 bytes (PAR 01D + PAR 02D)
			X'43'	TI = 24 bytes (PAR 01D + PAR 02D + PAR 03D)
2	TKL	Header length	X'00'	No meaning assigned
3	FST5	Function control char. 5 LA keys are blocked for user and also within P functions	Bit 7 6 5 4 3 2 1 Bit 7: v 1	
			Bit 1 → 0: / 1:	LA1 key 1)
			Bit 2 → 0: / 1:	LA2 key 1)
			Bit 3 → 0: / 1:	LA3 key 1)
			Bit 4 → 0: / 1:	LA4 key 1)
			Bit 5 → 0: / 1:	LA5 key 1)
			Bit 6 → 0: / 1:	LA6 and LA7 keys 1)

Char.	Desig.	Meaning	Coding	Explanation	
4	FST6	Function control char. 6 P keys blocked for user.	Bit 7 6 5 4 3 2 1 v 1		
			Bit 1	0: 1: P1 key	1)
			Bit 2	0: 1: P2 key	1)
			Bit 3	0: 1: P3 key	1)
			Bit 4	0: 1: P4 key	1)
			Bit 5	0: 1: P5 key	1)
			Bit 6	0: 1: P6 - P20 keys	1)
5	FST7	Function control char. 7 P key programming blocked	Bit 7 6 5 4 3 2 1 v 1		
			Bit 1	0: 1: P1 key	2)
			Bit 2	0: 1: P2 key	2)
			Bit 3	0: 1: P3 key	2)
			Bit 4	0: 1: P4 key	2)
			Bit 5	0: 1: P5 key	2)
			Bit 6	0: 1: P6 - P20 keys	2)

Char.	Desig.	Meaning	Coding	Explanation
6	SAW2	Send command 2 (for DÜ2 only)		SAW1 and SAW2 have the same range of functions. Invalid codes are treated as 'Send unprotected fields'.
			X'40'	Send unprotected fields
			X'41'	Send modified fields
			X'42'	Send unprotected fields from cursor
			X'43'	Send display buffer contents including all IS sequences
			X'44'	Send unprotected fields, without null characters
			X'45'	Send modified fields, without null characters
			X'46'	Send unprotected fields from cursor, without null characters
			X'47'	Reserved for field mode
			X'48'	Send from start marker to cursor, without null characters. The valid start marker is the one immediately preceding the cursor.
			X'4C'	Send unprotected fields with relevant null characters.
			X'4D'	Send modified fields with relevant null characters.
			X'4E'	Send unprotected fields from cursor, with relevant null characters.
			X'4F'	Send modified fields only, without null characters.
7	FST8	Function control char. 8 Badge information request	Bit 7 6 5 4 3 2 1 Bit 1 v v v v v v 1 0 0 0 0 0	0: Badge entry blocked 1: Badge entry expected (Keyboard entry blocked)
8	—	—	—	Reserved

1) 0: Use of key allowed
   1: Use of key disallowed
2) 0: Key programming allowed
   1: Key programming disallowed

Table 4-16 Description of the characters of PAR 01L

4.2.1.3 Format of parameter range PAR 00D

Terminal parameter range PAR 00D, which is down-line loaded from the host, is reserved for device functions initiated after completion of host-to-terminal transmission.
The parameter range can be loaded via parameter entries or via PARAMO of a message header. If a message header is used, the values relating to device function implementation are moved automatically from PAR 00L (GEF1) to PAR 00D.

Note

If intra-system data interchange is to be initiated by the host, the initiating function must be loaded in the allocated parameter range (PAR 10D - PAR 70D).

Character	Designation	Meaning
1	GEF1	Device function character 1
2	GEF2	Device function character 2
3	——	Reserved
4	LAP1	LA function parameter 1
5	AZL	Status line
6	——	Reserved
7	——	Reserved
8	——	Reserved

Table 4-17 Survey of the characters of PAR 00D

Char.	Desig.	Meaning	Coding	Explanation
1	GEF1	Device function char. 1	As per device function code tables (4-28 and 4-29) or identifier character of a three-char. sequence: X'20' (space).	The device function entered is initiated after end of host-to-terminal transmission. The following may be entered: 1. 2nd character of a two-character sequence (with initiation of corresponding device function). 2. Identifier character of a three-character sequence (X'20') - with initiation of corresponding device function by GEF2. The code extension char. ESC is not to be entered GEF=X'00' does not cause initiation of a device function. Device functions are implemented as for keyboard initiation.
2	GEF2	Device function char. 2	As per third character of a three-character sequence (Table 4-29).	The device function entered is initiated after end of host-to-terminal transmission. GEF2 is only interpreted if GEF1 contains the identifier character of a three-character sequence.
3	——	——	X'00'	Reserved
4	LAP1	LA function parameter.	X'40'	Transmission to and from the host is handled via the terminal screen. Device functions within a message are implemented.
			X'41'	Output to local devices (printers) is handled without modifying screen contents (see bypass mode).
5	AZL	Status line	X'40'	Operating status displayed
			X'41'	System line displayed
6	——	——	X'00'	Reserved
7	——	——	X'00'	Reserved
8	——	——	X'00'	Reserved

Table 4-18 Description of the characters of PAR 00D

4.2.2 Message Section Including Positioning Commands (IS4 Sequence)

The cursor can be positioned, by means of positioning commands within a message section, at any one of the 1920 terminal character positions (24 lines, 80 columns).
The message may contain any number of positioning command sequences in any order.

Format of a positioning command sequence:

The positioning command comprises

- the identifier character (IS4 = X'1C'),
- the base address for line (ZLA) and column (SPA),
- the page address (SAD).

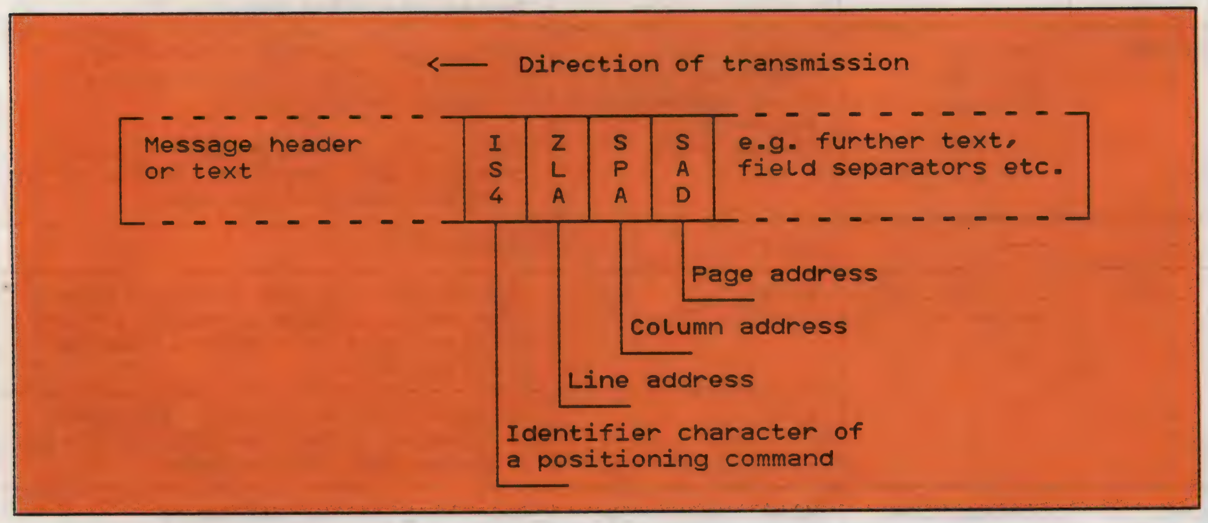


Fig. 4-3 Format of a positioning command

<u>Base address:</u> The base address, consisting of line and column address, indicates the position of the cursor on the screen.
Invisible field separators are located before the base address.

<u>Line address:</u> The coding of the line address can be determined by the status of bit 2 of character 5 (FST3) in PAR 00L when transmission is from terminal to host:

- Bit 2 = 0: Coding compatible with 8161 Data Display Terminal, versie B
- Bit 2 = 1: Coding <u>incompatible</u> with 8161 Data Display Terminal. versie A

The corresponding line address table allows for both types of coding.

<u>Page address:</u> <u>Only</u> SAD = X'30' is used on the 9749, 9750 and 9752 Data Display Terminals.

Coding of the line address

In the case of terminal-to-host transmission, both the coding of the line address (according to versions A or B) and the cursor position in PAR 01E (character 3) are determined by the status of character 5 of PAR 00L in the following way:

- Character 5, Bit 2 = 1: Coding for 9749, 9750 and 9752 is version A,
- Character 5, Bit 2 = 0: Coding for 9749, 9750, compatible with 8161 Data Display Terminal, is version B.

In the case of host-to-terminal transmission, there is no difference in the coding of the line address except by the WDH and LVA sequences. When the device functions WDH and LVA are output, the line address must be encoded in accordance with version A.

Line address					
Version A			Version B		
Line	Coding EBCDIC	Coding ISO-7	Line	Coding EBCDIC	Coding ISO-7
1	07	7F	1	F7	37
2	FF	7E	2	F6	36
3	FD	7D	3	F5	35
4	4F	7C	4	F4	34
5	FB	7B	5	F3	33
6	A9	7A	6	F2	32
7	A8	79	7	F1	31
8	A7	78	8	F0	30
9	A6	77	9	61	2F
10	A5	76	10	4B	2E
11	A4	75	11	60	2D
12	A3	74	12	6B	2C
13	A2	73	13	4E	2B
14	99	72	14	5C	2A
15	98	71	15	5D	29
16	97	70	16	4D	28
17	96	6F	17	7D	27
18	95	6E	18	50	26
19	94	6D	19	6C	25
20	93	6C	20	5B	24
21	92	6B	21	7B	23
22	91	6A	22	7F	22
23	89	69	23	5A	21
24	88	68	24	40	20

Table 4-19 Line address

Coding of the Column address

Column address

Column	Coding EBCDIC	ISO-7	Column	Coding EBCDIC	ISO-7	Column	Coding EBCDIC	ISO-7	Column	Coding EBCDIC	ISO-7
1	07	7F	21	92	6B	41	E6	57	61	C3	43
2	FF	7E	22	91	6A	42	E5	56	62	C2	42
3	FD	7D	23	89	69	43	E4	55	63	C1	41
4	4F	7C	24	88	68	44	E3	54	64	7C	40
5	FB	7B	25	87	67	45	E2	53	65	6F	3F
6	A9	7A	26	86	66	46	D9	52	66	6E	3E
7	A8	79	27	85	65	47	D8	51	67	7E	3D
8	A7	78	28	84	64	48	D7	50	68	4C	3C
9	A6	77	29	83	63	49	D6	4F	69	5E	3B
10	A5	76	30	82	62	50	D5	4E	70	7A	3A
11	A4	75	31	81	61	51	D4	4D	71	F9	39
12	A3	74	32	4A	60	52	D3	4C	72	F8	38
13	A2	73	33	6D	5F	53	D2	4B	73	F7	37
14	99	72	34	6A	5E	54	D1	4A	74	F6	36
15	98	71	35	BD	5D	55	C9	49	75	F5	35
16	97	70	36	BC	5C	56	C8	48	76	F4	34
17	96	6F	37	BB	5B	57	C7	47	77	F3	33
18	95	6E	38	E9	5A	58	C6	46	78	F2	32
19	94	6D	39	E8	59	59	C5	45	79	F1	31
20	93	6C	40	E7	58	60	C4	44	80	F0	30

Table 4-20 Column address

Coding of the page address

Data display terminals 9749, 9750, and 9752 use only SAD=X'30' to page. address.

Distance	Coding EBCDIC	ISO-7
0	F0	30

Table 4-21 Page address

4.2.3 Message section including field separators (IS2 and IS3 sequences)

Field separators define the attributes of a field.
The following characters are termed field separators

- Field handling characters (FBZ),
- Display control characters (ASZ).

Both field separators can be used independently.

Field separators

- are identified by IS2 or IS3 code extension characters,
- are always located at the beginning of a field,
- are entered in the terminal memory only during downline-loading from host,
- are located, invisible, between the displayed characters on the screen,
- do not restrict the number of displayable characters (80) per line,
- cannot be entered via the keyboard.

Up to 48 field separators can be inserted per line whereby not more than two field separators (1 FBZ, 1 ASZ) are allowed to follow one another.
If additional field separators are nevertheless inserted in a line containing 48 field separators, the following occurs:

- Field separators inserted after separator 48 are ignored.
- Field separators inserted before separator 48 cause always the last separator in the line to be eliminated.

The assignment of field handling characters (FBZ) and display control characters (ASZ) for video attribute differentiation are specified by

- switch 5, bit 4 of the setting memory or
- by means of a soldered strap on a character generator module

in various versions of data display terminals:

- switch 5, bit 4 = 0: Field handling characters and display control characters are assessed separately.
- switch 5, bit 4 = 1: Field handling characters are assigned fixed video attributes (see table 4-27).

The field separators recorded in the memory can be modified during an output sequence from the host. The field separator to be modified is addressed by means of a positioning command and

- overwritten with a new field separator or
- eliminated by being overwritten with text.

Format of character sequence with field separators

The character sequence required for formatting a message section with field separators comprises

- identifier characters (code extension characters)
  - IS2 code extension character for identifying field handling character (X'1E') or
  - IS3 code extension character for identifying display control character (X'1D'),
- field handling character (FBZ) or
- display control character (ASZ).

The identifier characters are not stored in the terminal memory.

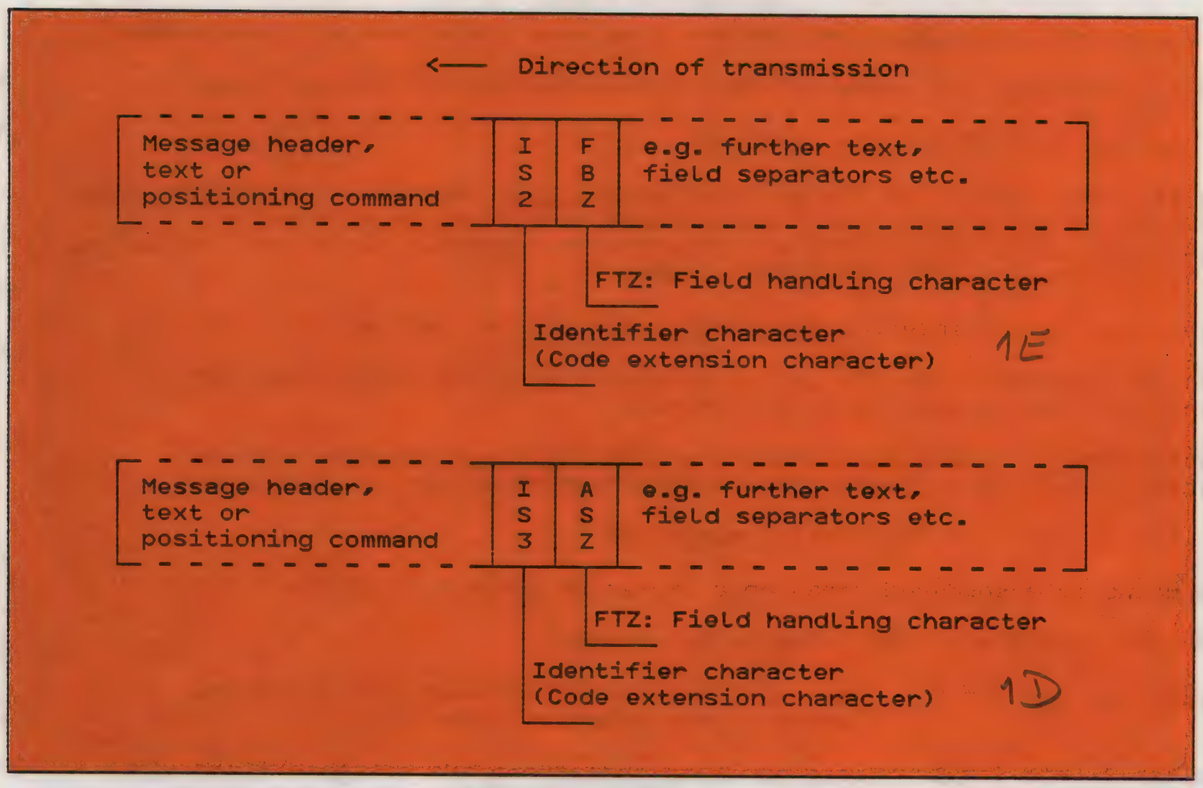


Fig. 4-4 Character sequence with field separators

Field handling characters (IS2 sequence)

Fields are formed exclusively by means of field handling characters (FBZ). Field attributes are determined by a <u>single</u> field handling character and remain valid until the <u>next</u> field handling character is encountered. Field handling characters can be used independently of display control characters (ASZ).

The following field attributes are determined by field handling characters:

Coding	Field attribute	Desig.	Explanation
Bit 7 6 5 4 3 2 1	depending on bit 6 and bit 1:		
Bit 7: 1; Bit 6: 0, Bit 1: 0 —>	=00: variable field	V	Fields can be written freely.
Bit 6: 0, Bit 1: 1 —>	=01: protected, non-sendable	P	Field contents cannot be modified via the keyboard. They are not included in transmission to host if the fields are defined as protected and non-sendable unless SAW=X'4F' and no MOD bit is set.
Bit 6: 1, Bit 1: 0 —>	=10: protected field, sendable	PS	Field contents cannot be modified via the keyboard. They are included in transmission to host unless SAW=X'4F' and no MOD bit is set.
Bit 6: 1, Bit 1: 1 —>	=11: protected field, only the addresses sent	PA	Provided SAW=X'41', X'45', X'4D'
Bit 2 —>	=0: alphanumeric field	AN	Entry via the keyboard of all displayable characters possible: numerics, alphabetics (large and small), special characters.
Bit 2 —>	=1: numeric field	N	Only numerics and the following special characters * + , - . / may be entered via the keyboard.

Bit 3 —>	=1: modified field	MO	Modified fields cannot be controlled by programming means. They are created by
			- host action,
			- keyboard entry of
			  - valid character,
			  - insert/delete of characters,
			  - LFZ function (depending on status of FST2 of PAR 00L),
			- marking of markable fields.

Bit 3 —>	=0:		MOD bit of FST2 (PAR 00L) is reset - after each transmission to/from host, - by output of FAZ in message, - via LZF and LVD functions, - by repeated marking.

Coding		Field attribute	Desig.	Explanation
Bit 4	=0:	non-markable field	NM	The field cannot be marked, either by a light pen or via the keyboard.
	=1:	markable field	MA	The field can be marked via the keyboard or with a light pen. The field can be visibly marked (flashing) only in conjunction with a display control character (X='40') which means that two field separators must be located at the beginning of a markable field.
Bit 5	=0:	non-printable field	NP	The field cannot be output to a printer.
	=1:	printable field	PF	The field can be output to a printer.

Table 4-22 Field handling characters

Coding of the field handling characters

Field attribute	Coding EBCDIC	Coding ISO-7	Field attribute	Coding EBCDIC	Coding ISO-7
V,NM,NP,A	7C	40	PS,NM,NP,A	4A	60
V,NM,NP,N	C2	42	PS,NM,NP,N	82	62
V,NM,PF,A	D7	50	PS,NM,PF,A	97	70
V,NM,PF,N	D9	52	PS,NM,PF,N	99	72
V,MA,NP,A	C8	48	PS,MA,NP,A	88	68
V,MA,NP,N	D1	4A	PS,MA,NP,N	91	6A
V,MA,PF,A	E7	58	PS,MA,PF,A	A7	78
V,MA,PF,N	E9	5A	PS,MA,PF,N	A9	7A
G,NM,NP,A	C1	41	PA,NM,NP,A	81	61
G,NM,NP,N	C3	43	PA,NM,NP,N	83	63
G,NM,PF,A	D8	51	PA,NM,PF,A	98	71
G,NM,PF,N	E2	53	PA,NM,PF,N	A2	73
G,MA,NP,A	C9	49	PA,MA,NP,A 1)	89	69
G,MA,NP,N	D2	4B	PA,MA,NP,N	92	6B
G,MA,PF,A	E8	59	PA,MA,PF,A	A8	79
G,MA,PF,N	BB	5B	PA,MA,PF,N	FB	7B

1) Combinations with PA are not partical

Table 4-23 Coding of field handling characters

Display control characters (IS3 sequence)

A <u>single</u> display control character determines the video attributes until the <u>next</u> display control character takes over.
Display control characters can be used independently of field handling characters.
The following video attributes are determined by the display control characters:

Coding	Video attribute	Desig.	Explanation
Bit 7 6 5 4 3 2 1			
Bit1	=0: No flashing	NF	
	=1: Flashing	FL	Characters flash at a frequency of around 2 Hz
Bit2	=0: Normal types	SF	
	=1: Underscored or reverse	US	Characters are underscored or displayed in reverse (changeover via jumper).
Bit3	=1: Blanking	BL	Characters controlled invisibly
Bit4	=0: Normal intensity	NL	
	=1: Reduced intensity	RI	Characters are displayed with reduced intensity. Contrast (normal(reduced intensity) is adjustable

Table 4-24 Display control characters

Video attributes on the 9752 Data Display Terminal

The following specific video attributes are to be noted with regard to 9752 Data Display Terminal:

Video attributes 9749, 9750	9752
Reduced intensity	Yellow
Normal intensity	Green
Flashing	Flashing and green
Flashing and reduced intensity	Flashing and yellow
Underscored	Red
Flashing and underscored	Flashing and red
Reduced intensity and underscored	White
Flashing, reduce intensity and underscored	Flashing and white
Blenking	Blanking

Table 4-25 Specific video attributes of the 9752 Data Display Terminal

Coding of the display control characters

Video (display) attribute 9749, 9750	9752	Coding EBCDIC	ISO-7
NI, SF, NF	green	7C	40
NI, SF, FL	green, flashing	C1	41
NI, US, NF	red	C2	42
NI, US, FL	red, flashing	C3	43
RI, SF, NF	yellow	C8	48
RI, SF, FL	yellow, flashing	C9	49
RI, US, NF	white	D1	4A
RI, US, FL	white, flashing	D2	4B
BL 1)	dark (blank)	C4	44

1) Attribute combinations with blanking are not practical

NF Non-Flashing
FL Flashing
SF Standard Font
US Underscore
BL Blanking
NI Normal intensity
RI Reduced intensity

Table 4-26 Coding of the display control characters

4.2.3.1 Field Attributes in Conjunction with Assigned Display Attributes

Particular field attributes can be combined with certain video attributes. This mode allows video attribute control to be undertaken by the field character as well. No display control characters are required here. The maximum number of field separators allowed per line (48) are, therefore, available wholly for the field handling characters. The use of display control characters is still valid, however the display control character must be located immediately preceding the character to be defined or at the start of the field. Of course the maximum possible number of fields per line is reduced in this way.
This operating mode - field attributes in association with video attributes - can be used subject to the data display terminal version (see table 2-1).

Read-only memory:
This mode is always ready to use.

Setting memory:
This mode can be activated using the S5 switch.

The assignment of video attributes to field attributes (by selection) in the display terminal can be hardware-implemented by means of soldered straps.

The following attribute assignments have been set ex-works whereby alterations are possible by DP maintenance personnel.

Bit	Value	Field attribute	Assignment	Display attribute
1	1	Protected field, non-sendable to host (Bit 6 = 0).		Normal intensity
2	1	Numeric field		Flashing
3	1	Modified field		
4	1	Markable field		Underscore/reverse vid. (jumper enables changeover)
5	1	Printable field		
6	1	Protected field, sendable to host (Bit 1 = 0).		Blanking
	1	Protected field, only field addresses are sent (Bit 1 = 1).	....	Reduced intensity
7	1	For avoiding transmission control characters		

——— Ex-works, inalterable

.... Ex-works, alterable by DP maintenance personnel

Table 4-27 Attribute assignment

Up to five field attributes can be combined with video attributes on a one-to-one basis.
Field attributes are not allowed to combine with one another although the video attributes have this possibility.

Note

Attention is to be paid to the specific video attributes in the case of the 9752 Data Display Terminal (see table 4-25)!

4.2.3.2 Modification of Field Separators

Field separators can be modified by

- insertion,
- deletion,
- overwriting.

The modification is handled in an output sequence from the host.

The message can contain

- field separators only,
- text and field separators,
- text only (all field separators deleted).

4.2.4 Message Section, Including Device Functions (ESC Sequences)

An abundance of device functions allow the programmer generous scope by the formation of the screen contents and facilitate the operator's work at the terminal.

The following device functions can be implemented:

- cursor positioning functions,
- text shift and editing functions,
- erase functions,
- send functions,
- intra-system data interchange functions,
- programmable key functions,
- special functions.

Device functions can be initiated both by the keyboard and downline via the host.
However, the restrictions noted in the following tables are to be observed because, for example, some of the functions can be initiated only during output from host.
Device functions can be entered in the

- message header,
  - by means of values in the message header,
- message section,
  - by means of values in the parameter entries (PAG),
  - by means of entries in the text.

Within the text, the device functions are identified by means of identifier characters (code extension characters).

Limitation

The following sequences can be recorded in the parameter entries:

- in PAR 00L, only two-character sequences (GEF1),
- in PAR 00D, two-character and three-character sequences (GEF1 and GEF2).

The device functions LVA and WDH can be entered only in the text section because positioning commands are necessary.

Device control characters for

- send functions (DÜx, Kx, Fx),
- intra-system data interchange functions (LAx) and
- programmable key functions which contain one of the previously mentioned send functions,

can be entered only in the parameter entries in character 4 (GEF1) of parameters PAR 00L and PAR 00D.
If these functions are nevertheless entered into a text section, as an ESC sequence, they are ignored. The terminal sets, in addition, an error message in character 6 (FEM) of PAR 01E.

Implementation of device functions

Device functions, when they are recorded in the parameter entries, can be implemented only after a completed text transmission.
When the device functions are entered as an ESC sequence within the text block of a message section, the respective function is implemented immediately and during the continuous text display on the screen.
In the case of transmission from terminal-to-host, the code of the send key actuated (DUx, Kx, Fx) is entered in character 7 (CDS) of PAR 00E.

Format of the character sequence, including device functions, within the text

The character sequence comprises

- the identifier character (code extension character) of a two-character sequence:
  ESC (X'1B') or
- the identifier characters (code extension characters) of a three-character sequence:
  (X'1B20') and
- the device functions (see tables 4-28 and 4-29).

Exception

In the case of the device functions LVA and WDH, the device function is followed by a positioning command with the line address and the column address.

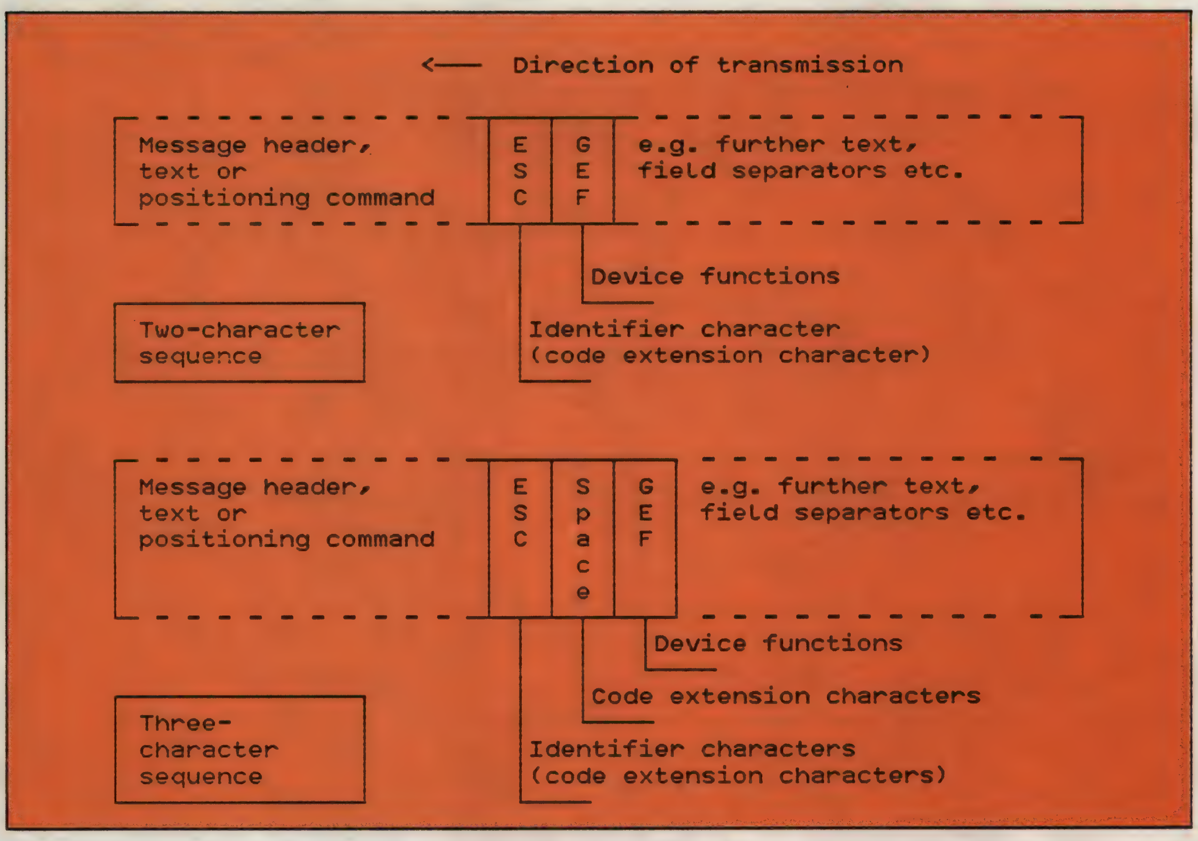


Fig. 4-5 Character sequences with device functions

Coding of the device functions:

	0	1	2	3	4	5	6	7	
0				AKA	SMR	PAR	RU	BRS	0
1					SML	LA1	P2	LZE	1
2					SMO	LA2	SBA		2
3					SMU	K1	P1		3
4					SNZ	K2	LSP		4
5					SZA	K3	LVD		5
6					TAR	K4	DÜ1		6
7					TAL	K5			7
8					EFG		P3		8
9		EM			AFG		P4		9
A			P5	K14	AFZ		P		A
B		ESC	P6	K13	EFZ	F1	SDZ		B
C				K12	LZF	F2	MAR		C
D				K11	K6	F3	FAZ		D
E				K10	K7	F4	RS		E
F				K9	K8	F5	VA		F
	0	1	2	3	4	5	6	7	

Table 4-28 Device funtions of the two-character sequence

<u>Example</u>

Cursor to start of screen (SBA): ESC-GEF = X'1B62' (ISO-7-Bit-Code).

	0	1	2	3	4	5	6	7	
0			SP		P7	AM	FON		0
1					P8	DU2	PAK		1
2					P9	SS	WDH		2
3					P10		LVA		3
4					P11				4
5					P12				5
6					P13				6
7					P14				7
8					P15				8
9					P16				9
A					P17				A
B		ESC			P18	LA3			B
C					P19	LA4			C
D					P20	LA5			D
E						LA6			E
F						LA7			F
	0	1	2	3	4	5	6	7	

Table 4-29 Device functions of the three character sequence

Example

Initiation of the P 20 function: ESC-Space-GEF = X'1B204D' (ISO-7-Bit-Code).

4.2.5 Description of the Device Functions

All of the following device functions can be initiated downline via the host and to a great extent also via the keyboard.

Cursor functions

The cursor functions allow the cursor to be moved to all positions on the screen.

Function		Coding		Implementation	
		EBCDIC	ISO-7	Non-formatted mode	Formatted mode
SMR	Cursor right	7C	40	Cursor jumps one char. forwards. At end of screen, cursor jumps to start of screen.	If autotab (ATAB) is activated and bit 4 of FST1 in PAR 00L is 0, the cursor over protected, non-markable fields, automatically moving to the first column of the next or preceding unprotected field.
SML	Cursor left	C1	41	Cursor jumps one char. backwards. At start of screen, the cursor jumps to end of screen.	
SMO	Cursor up	C2	42	Cursor jumps one line upwards, remaining in the same column. Upon reaching line 1, the cursor is positioned in line 24 in same column.	If autotab (ATAB) is activated and bit 4 of FST1 in PAR 00L is 1, the cursor is positioned in accordance with the function initiated, irrespective of field definition.
SMU	Cursor down	C3	43	Cursor jumps one line downwards, remaining in the same column. Upon reaching line 24, cursor is positioned in line 1.	
SNZ	Cursor to start of next line	C4	44	Cursor jumps to start of next line below. Upon reaching line 24, the cursor jumps to start of screen.	
SZA	Cursor to start of line	C5	45	Cursor jumps to start of line.	

Function		Coding		Implementation	
		EBCDIC	ISO-7	Unformatted mode	Formatted mode
SBA	Cursor to home position	82	62	Cursor skips to the start of screen.	
SDZ	Cursor to start of line above	92	6B	Cursor jumps to column 1 of line above. On reaching line 1, the cursor skips to the start of line 24.	
TAR	Tab right	C6	46	Cursor skips forward 16 positions at a time to columns 1, 17, 33, 49 and 65 of a line. On reaching column 65 of line 24, the cursor jumps to the start of screen.	Cursor skips forward to column 1 of the next modifiable field.
TAL	Tab left	C7	47	Cursor skips back 16 positions at a time to columns 65, 49, 33, 17 and 1. On reaching the start of screen, the cursor jumps to column 65 of line 24.	Cursor jumps back to column 1 of the next modifiable field.

Table 4-30 Cursor functions

Text Shifting Functions

Function		Coding		Implementation	
		EBCDIC	ISO-7	Unformatted mode	Formatted mode
EFG	Insert charac- ters	C8	48	Characters are moved forwards, from cursor position, by the number of charac- ters inserted. Null characters in the text are over- written. If there are no more null characters left, all subsequent chars. are moved forwards. The function remains effective beyond the end of the line. Characters shifted beyond the end of the screen are lost. If, during insertion, a LZE (Logical end of line) character reaches the end of a physical line, the characters in this and the following lines are moved forwards until the conditions for logical end of line are met. Insert characters mode is deactivated via the RS key.	As for unformatted mode, but restricted to an unprotected field. Characters shifted beyond the end of a field are lost. Display control characters are not affected. Inserting a charac- ter via the key- board causes the MOD bit to be set in the field handling character.
AFG	Delete charac- ters	C9	49	The character at the cursor position is deleted and subse- quent characters are shifted backwards one column. Positions becoming free are occupied by null characters. The function is effective up to end of screen.	As for unformatted mode, except that the function is not effective beyond the end of an unprotec- ted field. Deletion via the keyboard causes the MOD bit to be set in the field handling character.

Function		Coding		Implementation	
		EBCDIC	ISO-7	Non-formatted mode	Formatted mode
EFZ	Insert line (can be blocked for user via char. FST1 of PAR 00L).	D2	4B	If bit 2 of FST1 in PAR 00L is 0, all lines below and including the line the cursor is in are shifted downwards. If bit 2 of FST1 in PAR 00L is 1, same as above applies. In addition, the last line is copied to the cursor line.	As for non-formatted mode. Format data is treated as follows: When bit 2 of FST1 in PAR 00L is 0, the field separators and protected data from the last line are copied to the cursor line and unprotected data replaced by null characters. When bit 2 of FST1 in PAR 00L is 1, field separators and all data from the last line are copied to the cursor line.
AFZ	Delete line (can be blocked for user via char. FST1 of PAR 00L).	D1	4A	If bit 3 of FST1 in PAR 00L is 0, all lines below and including the line the cursor is in are shifted upwards. If bit 3 of FST1 in PAR 00L is 1, the above applies. In addition, the last line is copied to the cursor line.	As for non-formatted mode. Format data is treated as follows: When bit 3 of FST1 in PAR 00L is 0, the field separators and protected data are copied from the cursor line to the last line. When bit 3 of FST1 in PAR 00L is 1, field separators and all data in the cursor line are copied to the last line.
RU	Roll-up function (can be blocked for user via char. FST1 of PAR 00L).	4A	60	Lines 2 - 24 are each moved one line upwards. The chars. in line 1 are lost. Null characters are inserted in line 24. Cursor is positioned at start of line 24.	As for non-formatted mode. Field separators are also moved. Line 24 contains no field separators and those in line 1 are lost.

Function		Coding		Implementation	
		EBCDIC	ISO-7	Non-formatted mode	Formatted mode
RUB	Roll-up mode (activated and de-activated via char. FST1 in PAR OOL).	—	—	After the last write for line 24 has been performed, lines 2 - 24 are each moved one line upwards. The characters in line 1 are lost. Null characters are inserted in line 24. Cursor is at start of line 24. If a logical end of line (LZE) is entered in line 24, the rest of this line is overwritten with null characters, the roll-up function is initiated and the cursor is positioned at the start of the new line 24. Once 23 lines have been moved upwards, the RU function is disabled. It is not enabled until the RS key has been depressed. The line counter is reset as a result of a dialog, too.	As for non-formatted mode, including all field separators. Note If RU mode and autotab (ATAB) are used, the last character on the screen must be in an unprotected field.

Table 4-31 Text shift and editing functions

Erase functions

<table>
<tr><th colspan="2" rowspan="2">Function</th><th colspan="2">Coding</th><th colspan="2">Implementation</th></tr>
<tr><th>EBCDIC</th><th>ISO-7</th><th>Non-formatted mode</th><th>Formatted mode</th></tr>
<tr><td rowspan="2">LSP</td><td>Clear memory (host-initiated in PAR 00D or PAR 00L (GEF1) or key-board-initiated via LSP key, which can be blocked via FST2 in PAR 00L.</td><td rowspan="2">84</td><td rowspan="2">64</td><td>Erased as follows: From start to end of screen (overwrite with null charac-ters), PAR 00L, PAR 10L - PAR 70L (depending on status of FST2 in PAR 00L).</td><td>As for non-formatted mode, including, nevertheless, all field separators.</td></tr>
<tr><td>LSP is initiated via ESC seq. in text.</td><td>Erase from cursor to end of screen (no repositioning of cursor). Parameter ranges are not erased.</td><td>As for non-formatted mode, including nevertheless, all field separators.</td></tr>
<tr><td rowspan="2">LVD</td><td rowspan="2">Erase variable data</td><td rowspan="2">85</td><td rowspan="2">65</td><td>Erase from cursor up to and including an end marker or to end of screen (overwrit-ten with null cha-racters). After com-pletion of the erase function, the cursor is in its original position.</td><td>Erase unprotected fields from cursor up to and including an end marker or to end of screen (over-written with null characters). After completion of the erase function, the cursor is in its original position.</td></tr>
<tr><td colspan="2">The erase limits are specified by bit 6 of character FST2 in PAR 00L as follows:
Bit 6=0: Erase from start of screen to end marker.
Bit 6=1: Erase from start to end of screen.</td></tr>
</table>

Function		Coding		Implementation	
		EBCDIC	ISO-7	Non-formatted mode	Formatted mode
LZF	Erase to end of line or end of field	D3	4C	Erase from cursor to end of line or to end marker in this line (the end marker also being erased). After the erase function, the cursor is in the original position.	All characters in an unprotected field are erased from cursor to end of field or up to and including an end marker in the same field. The cursor position is unchanged. The status of modify and flash bits is specified in bit 5 of character FST2 in PAR 00L.
LVA	Erase variable data up to address	83	63	All variable data between the cursor position and the address specified are erased. The position address in the second sequence specifies the position of <u>the first character that is not to be erased</u>. If the position address is missing, the next two characters following the LVA function will be used as the position address. If no address is specified or if specified and invalid, the function will not be performed. Modify (MOD) bits remain unchanged. <u>Note</u> The LVA function can be entered only in the text section because positioning commands are required. The positioning commands for the line address must contain values of table 4-19, version A. Identifier characters (IS4) and page address (SAD) entries <u>are omitted</u>. <u>Example</u> The variable data from start of screen up to line 10, column 1 is to be erased: ESC-SBA-ESC-SP-LVA-ZLA-SPA = X'2782274083A507' (EBCDIC coding).	

Table 4-32 Erase functions

Send Functions

Function		Coding EBCDIC	Coding ISO-7	Implementation: Unformatted mode / Formatted mode
DÜ1	Send initiated	86	66	When a send function is initiated, a send request (SAN) is sent to the host. SAN is displayed in line 25. SAN can be reset via the RS key. Message transmission is in accordance with the entries in character 1 (SAW1) in parameter range PAR 00L. The code of the send key used is entered in character 7 (CDS) in PAR 00E.
DÜ2	Send initiated	D8	51	Implementation as for DÜ1. Message transmission is in accordance with the entries in character 6 (SAW2) in parameter range PAR 01L. The code of the send key used is entered in character 7 (CDS) of PAR 00E.
K1	Send short message (Telegram) via keyboard or host (Kx).	E2	53	Initiation via keys K1 - K3
K2		E3	54	
K3		E4	55	
K4		E5	56	Initiation via ESC key and one of the keys on the right: V - K4
K5		E6	57	W - K5
K6		D4	4D	M - K6
K7		D5	4E	N - K7
K8		D6	4F	O - K8
K9		6F	3F	? - K9
K10		6E	3E	> - K10
K11		7E	3D	= - K11
K12		4C	3C	< - K12
K13		5E	3B	; - K13
K14		7A	3A	: - K14
				When a K function is initiated, the corresponding code is entered in character 7 (CDS) in PAR 00E. A short message (code) may comprise PAR 00E, PAR 01E and a text ID. A K function can also be initiated if the keyboard is blocked. If a K function is to be initiated by the host, the appropriate code must be entered in character 1 (GEF1) in PAR 00D.

Function		Coding		Implementation	
		EBCDIC	ISO-7	Unformatted mode	Formatted mode
F1	Send	BB	5B	Initiated via keys F1 - F3	
F2	via	BC	5C		
F3	funct. Fx.	BD	5D		
F4		6A	5E	Initiated via the ESC key and one of the keys on the right: When F function is initiated, the code complying with the latter is entered in character 7 (CDS) of PAR 00E and sent together with the displayed message. If an F function is to be initiated from the host, the appropriate code must be entered in character 1 (GEF1) in PAR 00D.	
F5		6D	5F		

Table 4-33 Send functions

Intra-system Data Exchange Functions

Function		Coding		Implementation	
		EBCDIC	ISO-7	Unformatted mode	Formatted mode
LA1	Local Initiation	D8	51	Depending on the address specifications in parameter ranges PAR 10L/D through PAR 70L/D, an output device attached locally LA5 - LA7 are initiated via ESC sequence to the data display terminal or a printer terminal connected to the same cluster controller will be selected for data exchange.	
LA2		D9	52		
LA3		BB	5B		
LA4	LA1 - LA4 are initiated directly via keyboard <u>and</u> LA5 - LA7 are initiated via ESC sequence <u>or</u> LA1 - LA7 are initated by the host	BC	5C		All printable fields from the cursor position to an end marker or to end of screen are printed out. Printer control characters in pintable fields are interpreted in accordance with the specifications in the print command (PAR 10L/D - PAR 70L/D)
LA5		BD	5D		
LA6		6A	5E		
LA7		6D	5F		

Table 4-34 Intra-system data exchange functions

Programmable key functions

Function		Coding EBCDIC	Coding ISO-7	Implementation Non-formatted mode	Implementation Formatted mode
P	Activate programm-ing mode	91	6A	The P function activates programming mode, in which the P registers can be programmed. If this mode is not activated, the P registers cannot be programmed. After initiation, 'TASTE Pn=?' (KEY Pn=?) is displayed in the terminal status line. After selecting the Pn function required, the corresponding register range is displayed and released for <u>programming</u>, provided no <u>programming inhibit</u> flag is set in character 5 (FST7) in parameter range PAR 01L. <u>The programming mode</u> is de-activated by initiating the P function again. The procedure for the downline-loading from the host is identical to keyboard loading.	
P1	Select P register (1 - 20)	83	63	If programming mode has been activated, the selection of one of the P function keys (P1 - P20) or an appropriate ESC sequence causes the range required to be displayed. The cursor is positioned at the start of the register range. <u>Storage capacity and entry capability:</u> Up to 16 characters can be entered in a P register. Storable characters are: alphanumerics, device control characters and field separators (e.g. ESC, IS2, IS3 and IS4 sequences). The characters are not interpreted during programming mode. If an entry has to be corrected, a null character can be keyed in the key being back-spaced one position. <u>Chaining of P functions:</u> In order to chain P functions, the code of a different P function may be entered in the P register already selected. However, this P function must have a higher value (=x) than the P register to be programmed. After a valid P function has been entered, programming is terminated and the register range exchanged for a new one.	
P2		81	61		
P3		88	68		
P4		89	69		
P5	*1	5C	2A		
P6		4E	2B		
P7		7C	40		
P8		C1	41		
P9		C2	42		
P10		C3	43		
P11	*2	C4	44		
P12		C5	45		
P13		C6	46		
P14		C7	47		
P15		C8	48		
P16		C9	49		
P17		D1	4A		
P18		D2	4B		
P19		D3	4C		
P20		D4	4D		

*1: 27..

*2: 2720..

Function	Coding		Implementation	
	EBCDIC	ISO-7	Non-formatted mode	Formatted mode
			Warning The DÜ function should be present only once. Keyboard initiation: When one of the P1 through P20 functions is initiated, the contents of the respective parameter range is interpreted. A user inhibit can be entered in character 4 (FST6) of PAR 01L as follows: - specific user inhibit for P1 - P5 individually, - general user inhibit for P6 - P20 as a whole. Note If the register range selected contains a send function, bit 5 of character 5 (FST3) of PAR 00L should have the value 1. Programming inhibit: If the P register is reserved for the host, activating a Px function key will cause 'DVA PN' to be displayed and the entry of characters inhibited. The characters entered by the host are processed. A programming inhibit can be entered in character 5 (FST7) of PAR 01L as follows: - specific inhibit for P1 - P5 individually - an inhibit for P6 - P20 as a whole. Host initiation: Host initiation via device control characters in the message or in PAR 00D. If the P register contains a function that is invalid during host-to-terminal transmission e.g. DÜ, K, F or LA, this function is not interpreted and the implementation of the P function continues.	

Table 4-35 Programmable key functions

Function		Coding EBCDIC	ISO-7	Implementation Non-formatted mode	Formatted mode
FAZ	Set field separators to initial state	94	6D		The MOD bits (= Bit 3) in all field handling characters and the flash bits (= Bit 1) in the display control characters of the marked fields are erased. An entry in character 3 (FST2) of PAR 00L indicates whether this function be initiated either upon reception of a device function command only or automatically after each output from the host.
MAR	Mark field	93	6C	A MAR entry is ignored because the function is effective in formatted mode only.	Markable fields start with field handling characters (Bit 4 = 1) and display control characters. A field can be marked if it contains the cursor and the MAR function is initiated (MOD bit in field separator being set). The marked field begins flashing as a visual acknowledgement. Each subsequent marking revokes the function for protected fields or restarts it again. Unprotected fields remain marked. Depending on the status of character 3 (FST2) Bit 6, PAR 00L, the flash bits are reset in those fields which are marked by a display control character by an output from host.

Function		Coding EBCDIC	ISO-7	Implementation: Non-formatted mode / Formatted mode
PAR	Modify parameter range	D7	50	Can be reset by initiating the RS function or an LA function. Note A message must not contain a PAR function.
RS	Reset	95	6E	Resets error flags and function statuses without modifying memory contents. Send requests set by acknowledgements cannot be reset by means of the RS function.
VA	Clear connection	96	6F	This function is effective for stand-alone display terminals only. After completion of a DÜ,P or LA function, the connection is cleared by the terminal.
AM	Start marker	D7	50	Unprotected fields, including null characters, are sent, providing the appropriate SAW is set, from and including the start marker through to and including cursor position. The start marker is interpreted only during terminal-to-host transmission. X'48' must be specified in char. 1 (SAW1) of PAR 00L or in char. 6 (SAW2) of PAR 01L. The last start marker preceding the cursor is always interpreted. If the start marker is missing, sending begins from start of screen. The start marker is ineffective at the end of screen.
EM	End marker	19	19	The end marker enables limiting of the following functions: - message printout, - erase operations, - terminal-to-host transmission if the appropriate send request (SAW1) is entered in parameter range PAR 00L. In the case of transmission from terminal-to-host, the end marker must be located in a sendable field.
BEL	Visual alarm	2F	07	Upon reception of the control character BEL, 'BEL' flashes in the status line and, if installed, an audible signal can be heard as well. BEL can be used, for example, to identify output from the host. BEL is reset using the BRS function. Note This function is programmable only as an individual text block without a message header or preceding the start or end sequence without STX...ETX.

Function		Coding EBCDIC	ISO-7	Implementation Non-formatted mode	Formatted mode
BRS	Reset BEL	97	70	Resets the alarm function BEL.	
AKA	Acoustic alarm	F0	30	If installed, an audible alarm is omitted when the control character AKA is received and 'BEL' is displayed. Example of application: AKA is activated at the end of a host-to-terminal transmission to indicate to the user that output is over. BEL is reset by means of the functions BRS or RS.	
SS	Set cursor	D9	52	A specification in character 5 (FST3) of PAR 00L, indicates whether, during host-to-terminal transmission, the cursor is to track the message as it is received, or remain at its current position. When the SS function is outputted, the cursor set at the current write position.	
WDH	Repeat	82	62	A character is repeated up to a specified line/column address. This repetition is equivalent to outputting the corresponding number of characters (e.g. field separators are overwritten). The function is effective from the cursor position to the screen line position specified by means of ZLA and SPA. If invalid addresses are specified, the function will not be implemented. If the specified address is before the current write position, the repeat function is implemented beyond the end of screen up to the address specified. If the specified address corresponds to the current write position, the entire display buffer is filled with pre-specified characters. Note The WDH function can be entered only in the text section as positioning commands are required. These commands must contain those line address values from table 4-19, version A. Identifier characters (IS4) and page address entries (SAD) are omitted. Example Line 1 is to be written with the letter 'A': ESC-SBA-ESC-SP-WDH-ZLA-SPA-A = X'2782274082FF07C1' (EBCDIC coding).	

Function		Coding		Implementation	
		EBCDIC	ISO-7	Non-formatted mode	Formatted mode
LZE	Logical end of line	98	71	Logical lines begin at start of line and end with LZE. They can be part of a line or several lines long. The cursor can reach all cursor positions. All characters after a LZE are erased up to end of line.	Logical lines begin at start of field and end with LZE. They can be part of a line or several lines long. The cursor reaches all cursor positions. All characters are erased up to end of field or end of line. Field separators are not erased.
FON	Form number	4A	60	This function is interpreted exclusively by the 8123 Printer and is treated by the terminal as a text character.	

Table 4-36 Special functions

4.3 FORMAT OF A SYSTEM-OUTPUTTED MESSAGE

The contrasted, bottom line (25th line) of the screen is termed, in the initial state (after power on) of the terminal, the device display line. Operating statuses are displayed in this line.

Depending on the value of character 5 (AZL) of PAR 00D, a system line can be displayed in line 25 in which operator prompting text and system messages can be stored.
These two alternatives are determined as follows:

- Character 5 (AZL) = X'40': Display of device display line.
- Character 5 (AZL) = X'41': Display of system line.

The system line is loaded with special parameter entries when a message from host is outputted.

128 characters can be stored:

- 80 (maximum) visible characters and
- 48 (maximum) invisible display control characters.

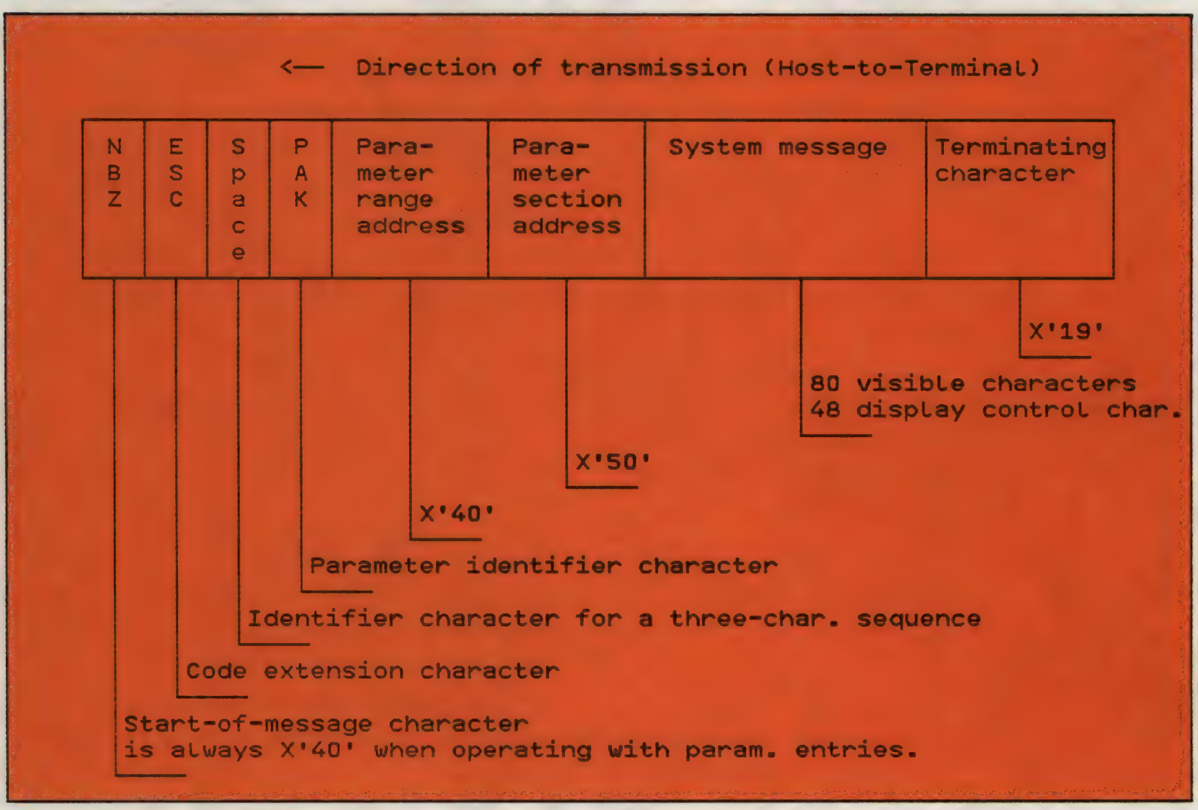


Fig. 4-6 Message format for the system line

The end marker (EM) is the terminating character for system messages and is non-displayable in the system line. Any part of the system line remaining, after the end marker, is erased with null characters.

IS2 and IS3 sequences are not interpreted in the system line.

If, during execution of a function, an error occurs or the P, PAR or EFG mode is activated, the system line is substituted automatically on the screen by the device status line. The system line is displayed again when the error condition is reset via RS or when the above-mentioned modes are deactivated.

Note

It is not practical to accompany a system line display with an audible alarm (AKA) or a visual alarm (BEL) as this would cause the system line to be substituted by the device display line.

4.4 MESSAGE FORMAT DURING OPERATION WITH A BADGE READER

The badge reader, which is integrated into different keyboard versions, enables files, that are protected either by the operating system or by user programs, to be accessed by entering user ID in the form of a badge.

4.4.1 Badge Formats

The badge reader can read badges with the following format:

- SIPASS and
- ABA.

The adjustment ex-works is for the SIPASS format but modification by means of a soldered strap in the keyboard enables conversion to ABA format.

4.4.2 Entering Badge Information

In order that badge information can be read and transferred, it is necessary to set the request bit in character 7 (FST8) of PAR 01L and display a request to enter on the screen.
The format and presentation of the enter request are specified by the operating system or the user program.
Keyboard entry is blocked once the request bit is set in PAR 01L.
An attempt to read without a prior request by the host causes the text 'GESPERRT' to be displayed in the device display line.

When a valid item of badge information is entered, bit 1 in character 5 (ZZ1) of PAR 00E is set to 1. At the same time, the display terminal resets the request bit in PAR 01L and the keyboard is released again.
Bit 1 of character 5 (ZZ1) in PAR 00E remains set until the badge is removed from the reader.

After the badge has been removed, the reader sends a K14 message to the host.

Entering faulty badge information

If there is no badge or the badge information is entered incorrectly, the read request from the host must be answered by a K14 message via the keyboard.
The K14 message is initiated via the ESC sequence.

The operating system or the user program can now reset the request bit in character 7 (FST8) of PAR 01L by outputting a message, thus releasing the keyboard.

4.4.3 Error Messages

The badge reader itself is unable to indicate errors.
Error messages are displayed on the terminal screen in the device display line.

Error message text	Cause of error	Error correction
GESPERRT (disabled)	Badge inserted in reader without request (AWL) from host.	Activation of RS key.
AUSWEISFEHLER (Badge error)	Badge was passed through reader - too slowly - at inconstant speed	Pass badge through reader again. If error text remains, enter 'ESC' and continue.
	Badge is defect Bade information invalid	Use a new badge.
	Badge format does not conform to keyboard setting	Adjust format setting
KARTE LEER (Badge empty)	Badge is blank	Use a new badge
	Badge inserted incorrectly	Pass badge through at correct angle.

Table 4-37 Error messages

5 DATA EXCHANGE WITH PRINTERS

A printer can be used for the output of data displayed on the screen of a data display terminal (e.g. local output) or the output of data not displayed (e.g. bypass operation).
Both the method of connection and the operating mode should be taken into account.
The diagram below shows the possible methods of exchanging data between data display terminal and printer.

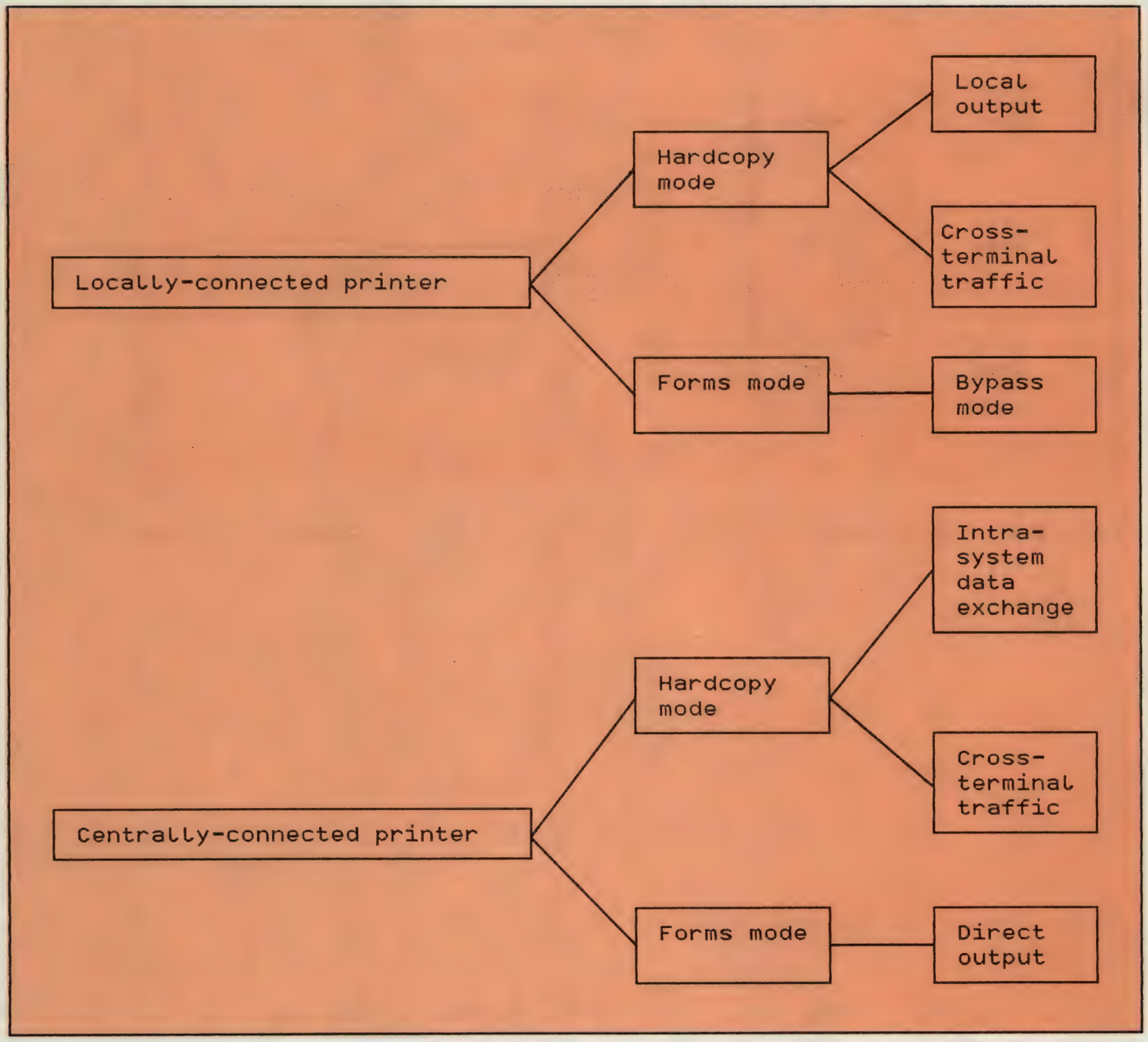


Fig. 5-1: Printer connection methods and operating modes

5.1 PRINTER CONNECTION METHODS

Printers may be connected either locally or centrally.

5.1.1 Local Connection

The expression 'locally-connected printer' is used to refer to a printer which is connected to the printer interface of a data display terminal.

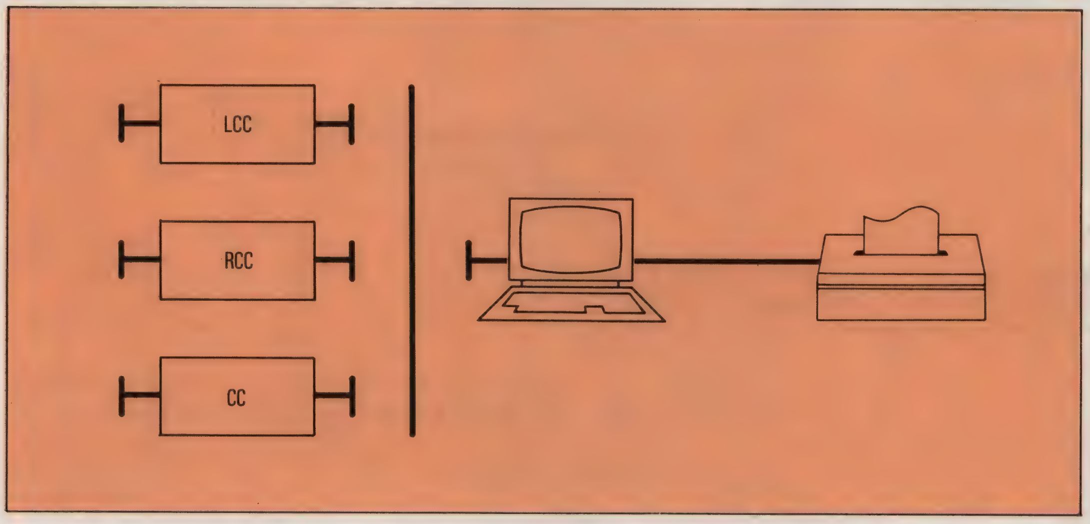


Fig. 5-2: Local connection of a printer to a data display terminal

5.1.2 Central Connection

Connection to 8112 Printer Terminal Controller

If a printer is connected to an 8112 Printer Terminal Controller, and the latter and the data display terminal are connected to the same

- 8170 Cluster Controller (Local) (LCC) or
- 8171 Cluster Controller (Remote) (RCC) or
- local BAM line controller (BAST) of a communication computer (CC)

then data from the data display terminal can be output on the printer.

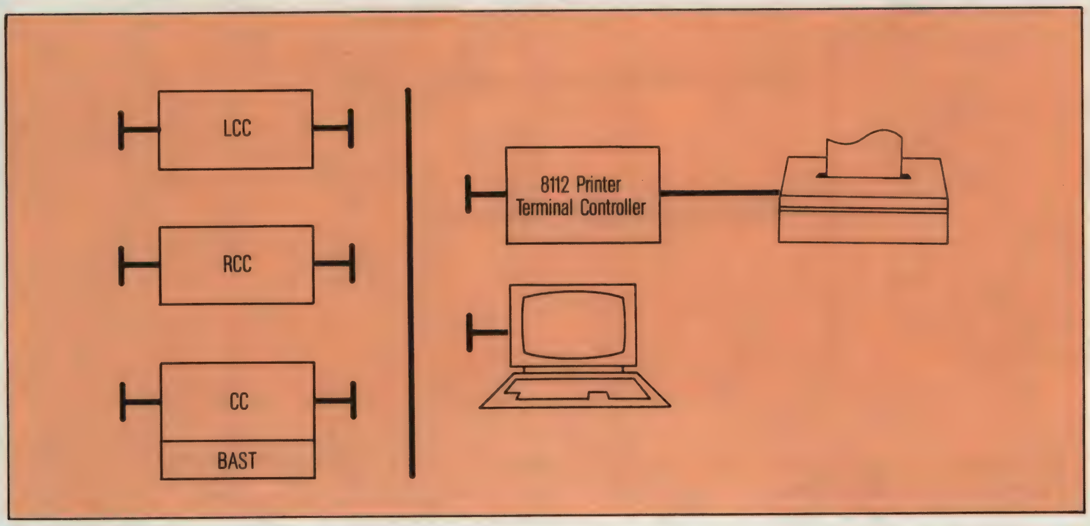


Fig. 5-3: Connection of a central printer to an 8112 Printer Terminal Controller

Connection via 90037-103 Integrated Printer Controller

If a printer with a 90037-103 Integrated Printer Controller and the data display terminal are connected to the same

- 8170 Cluster Controller (Local) or
- 8171 Cluster Controller (Remote) or
- Local BAM line controller of a communication computer

then data from the data display terminal can be output on the printer.

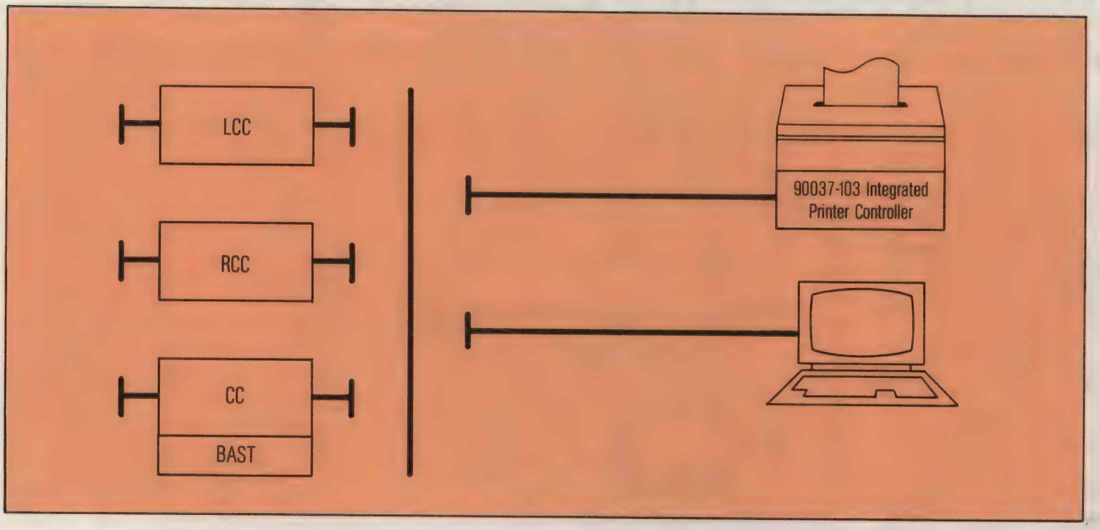


Fig. 5-4: Connection of a central printer with 90037-103 Integrated Printer Controller

5.2 PRINTER OPERATING MODES

There are basically just the following two operating modes for printers connected to data display terminals:

- hardcopy mode and
- forms mode.

The possible variations on these operating modes are described in the following sections.

5.2.1 Hardcopy Mode

The printer is assigned to a data display terminal:

- The local printer
  - by connection to a data display terminal or
  - via the function 'cross-terminal traffic' when connected to the communication computer.
- The central printer
  - via the function 'intra-system data exchange' or
  - via the function 'cross-terminal traffic' when connected to the communication computer.

In hardcopy mode the data displayed on the screen of a data display terminal is output on the printer in unmodified form.
The control characters (e.g. for positioning) contained in the text of the message are interpreted by the data display terminal. The data display terminal ignores any printer control characters which may be present, and does not pass them on to the printer.
The contents of the screen from the cursor to the end marker are output. If there is no end marker, everything up to the end of the screen is output.

In hardcopy mode the recipient of a message is always the data display terminal. The data displayed on the screen is additionally output on a printer.
If there is a malfunction in the local printer or in the printer terminal controller/printer controller of the central printer during data output, the data display terminal rejects new requests to receive from the computer (i.e. selection calls) with 'WABT' if the requests require a check to be made concerning the readiness of the printer.

If character 5 (FST3) of PAR 00L has bit 5=1 (dialog mode activated), then the keyboard will also be locked in the case of data output to a printer terminal controller/printer controller.
The keyboard can be released by sending a return message (acknowledgment), which must be requested by

- character 2 (PBH) of PAR 10L/D - 70L/D (local connection) or
- character 5 (ADA) of PAR 10L/D - 70L/D (central connection).

Printing can be initiated

- manually, by pressing an LAx key on the keyboard of the data display terminal. The functions LA1 - LA4 are assigned to the keys LA1 - LA4. Functions LA5 - LA7 can be initiated by ESC sequences. The corresponding parameter range PAR 10L - 70L is interpreted.

- automatically, by the computer itself. The device function effecting initiation must be entered in PAR 00D:

  - character 1 (GEF1) for LA1 - LA2 (2-character string)

  - character 1 (GEF1)
character 2 (GEF2) } for LA3 - LA7 (3-character string)

  Printing starts only <u>after</u> message transfer to the data display terminal has been completed.

<u>Note</u>

If the device function entered in PAR 00D is an LA function or a P function containing an LA function, then the corresponding parameter range PAR 10D - 70D is interpreted.
This parameter range must be prepared by the computer. The preparation can be performed in the same output, provided PAR 10D - PAR 70D comes <u>before</u> PAR 00D in the message.

Print output is always controlled from the data display terminal.
Message header, parameter ranges for the data display terminal and output message are thus set up <u>solely</u> for the data display terminal itself.

5.2.1.1 Local Output

Data exchange between a data display terminal and a printer locally connected to it is referred to as local output (local hardcopy mode). The data display terminal is the recipient of the message, which must therefore contain no printer control characters.

For local output, the following preconditions must be fulfilled:

- PAR 10L/D - 70L/D:
  - character 1 (KAN) = X'00' (output on local printer),
  - character 2 (PBH), with or without acknowledgment,
  - character 3 (GAD), device address (address set in the printer),
  - character 5 (ADA), bit 1=0 (hardcopy mode).
- PAR 00D (initiation via the host):
  - character 1 (GEF1) for LA1 - LA2 (2-character string)
  - character 1 (GEF1), character 2 (GEF2) } for LA3 - LA7 (3-character string)
  - character 4 (LAP1) = X'41' (bypass mode).

<u>Note</u>

When the data display terminal is switched on, default values are automatically loaded into parameter ranges PAR 10L/D - 70L/D of the terminal's data storage (bootstrap loading):

- KAN: X'00' (output on local printer),
- PBH: X'48' (without acknowledgment),
- GAD: X'31' in PAR 10L/D, X'32' in PAR 20L/D, etc.,
- GBF: X'53' (write command),
- ADA: X'40' (hardcopy mode).

These default values can be erased and overwritten by output of PAR parameter entries (SOM = X'40').

5.2.1.2 Cross-Terminal Traffic

This term is used to describe data exchange between

- a data display terminal (connected to a BAST local BAM line controller) and

- printers

which are all connected to the same communication computer (terminal computer, front-end processor, compact front-end processor, integrated front-end processor); see [8], [12], [26], [27].
Data exchange is controlled by a PDN system program [8].

Cross-terminal traffic can be initiated

- directly, by pressing an LAx key on the data display terminal keyboard or

- indirectly, by means of the application program linked to the initiating data display terminal.

Further details on the programming of cross-terminal data traffic may be found in the appropriate specialized publications [8], [12], [26], [27].

5.2.1.3 Intra-system Data Exchange

This term describes data exchange between

- data display terminals and

- central printers,

which are connected to the same 8170 Cluster Controller (local) or 8171 Cluster Controller (remote). Data exchange is controlled via the hardware/firmware.

The following preconditions must be fulfilled for intra-system data exchange:

- PAR 10L/D - 70L/D:

  - character 1 (KAN) ≠ X'00' (channel address = address of central printer, see Tables 5-7 and 5-8)

  - character 5 (ADA), bit 1=0 (hardcopy mode).

- PAR 00D:

  - character 4 (LAP1) = X'40' (no bypass mode).

The PDN operating system (for the 8171 Cluster Controller) and the the software in the host (for the 8170 Cluster Controller) remain unaffected by the intra-system data exchange function.

5.2.2 Forms Mode

The message must include all the control characters for the printer (e.g. CR, LF, etc.).

- Printer operation in which the printer is connected locally and the message is not displayed on the terminal screen is known as bypass mode.

- Printer operation in which the printer is connected centrally is known as direct output.

Printer output is controlled

- via the data display terminal in the case of local connection,

- via the 8112 Printer Terminal Controller or the 90037-103 Integrated Printer Controller in the case of central connection.

5.2.2.1 Bypass Mode

In the bypass mode the printer is always connected locally. Data exchange takes place without the message being displayed on the screen. The message must contain all the printer control characters.
After printing has been completed, the text transferred during the bypass operation is erased from the data display terminal's buffer storage.

The following preconditions must be fulfilled for bypass mode:

- PAR 10D - 70D:
  - character 1 (KAN) = X'00' (output to local printer),
  - character 2 (PBH), with/without acknowledgment,
  - character 3 (GAD), device address (address set in the printer).

- PAR 00D (initiation by the host):
  - character 1 (GEF1) for LA1 - LA2 (2-character string)
  - character 1 (GEF1)
character 2 (GEF2) } for LA3 - LA7 (3-character string)
  - character 4 (LAP1) = X'41' (bypass mode).

If a PAR 00D with character 4 (LAP1) = X'41' (bypass mode) is loaded during a current data output, then the message preceding the parameter entries (PAG) is processed normally and displayed on the screen.
Only the text following PAR 00D is not displayed on the screen, being printed out in a bypass operation.
The parameter ranges PAR 10D - 70D must therefore be loaded <u>before</u> PAR 00D.

Note

Bypass mode can only be initiated from the host.
In order to prevent the data display terminal being blocked for dialog applications, it is essential that acknowledgments be used when operating in the bypass mode.

5.2.2.2 Direct Output

The data display terminal takes no part in direct output.
The printer is connected centrally

- via an 8112 Printer Terminal Controller or
- via the 90037-103 Integrated Printer Controller

to a local cluster controller, remote cluster controller or communication computer.
For further details see [12].

5.2.2.3 Forms Mode with Video Display

In this mode the message (inclusive of all printer control characters) appears on the screen of the data display terminal. The control characters take up storage space in the display buffer of the data display terminal. Only certain printer control characters may be used [12].

The following preconditions must be fulfilled for this mode:

- PAR 10D - 70D:
  - character 5 (ADA), bit 1=1 (forms mode).
- PAR 00D:
  - character 1 (GEF1) for LA1 - LA2 (2-character string),
  - character 1 (GEF1)
    character 2 (GEF2) } for LA3 - LA7 (3-character string)
  - character 4 (LAP1) = X'40' (no bypass mode).

5.3 DATA EXCHANGE WITH LOCAL PRINTERS

In the case of local connection, output via the printer is only possible via the data display terminal, i.e. via the terminal's print buffer. A precondition for data exchange is that the necessary parameters are supplied for the parameter ranges PAR 00D and PAR 10L/D - 70L/D.

5.3.1 Format of Parameter Ranges PAR 10L/D - 70L/D

The parameter ranges PAR 10L - 70L must be supplied with the proper values if printing is to be initiated manually by pressing an LAx key on the data display terminal keyboard.
The parameter ranges PAR 10D - 70D must be loaded with the proper values if printing is to be initiated automatically by the host (characters 1/2, GEF1/2 of PAR 00D = LA1 - LA2 or LA3 - LA7).
The parameter values can be supplied via the same message (data output) provided PAR 10D - 70D precedes PAR 00D in the message.
Assignment of the device functions to the corresponding parameter ranges is shown in Table 4-10.

Note

It is only necessary to supply parameters for these parameter ranges if values other than the defaults loaded during bootstrapping are required.

Character	Designation	Meaning
1	KAN	Channel address
2	PBH	Parameter handling
3	GAD	Device address
4	GBF	Device instruction
5	ADA	Print command
6	RB1	Acknowledgment byte 1
7	RB2	Acknowledgment byte 2
8	——	Reserved

Table 5-1: Overview of characters in PAR 10L/D - PAR 70L/D

Char.	Design.	Meaning	Coding	Explanation
1	KAN	Channel address	X'00'	Addresses a printer locally connected to the data display terminal
2	PBH	Parameter handling	Bit 7 6 5 4 3 2 1 1 (bit 7), 1 0 0 (bits 4 3 2) Bit 6 = 0, Bit 1 = 0 → Bit 6 = 0, Bit 1 = 1 → Bit 6 = 1, Bit 1 = 0 → Bit 6 = 1, Bit 1 = 1 →	<u>Bits 6 and 1:</u> =00: } without acknowledgment =01: } (ACK) to host =10: positive ACK in the case of successful printing and negative ACK if print process is aborted =11: negative ACK only in the event of printing being aborted ACK requests are erased after initiation of the print operation
			Bit 5 →	=0: DDT outputs CR and LF to the printer. =1: DDT outputs only LF to the printer. Bit 5 is interpreted only in hardcopy mode, controlled by ADA or DDT end-of-line.
3	GAD	Device addr. Not interpreted by 9001 and 9004 Printers	X'31'	Local printer with GAD1
			X'32'	Local printer with GAD2
			X'33'	Local printer with GAD3
			X'34'	Local printer with GAD4
			X'35'	Local printer with GAD5
			X'36'	Local printer with GAD6
			X'37'	Local printer with GAD7
4	GBF	Device instruction	X'53'	Data output only (= write) possible.

Char.	Design.	Meaning	Coding	Explanation
5	ADA	Print command Not interpreted in bypass mode	Bit 7 6 5 4 3 2 1 Bit 1 1 0 Bit 2 0 0 0 0 0 1 0 1 0 0 1 1 1 0 0 1 0 1 1 1 0 1 1 1	=0: Hardcopy mode =1: Forms mode =0: Data output from the DDT without escape characters =1: The DDT inserts ESC3 at the beginning of a protected field and ESC4 at the start of an unprotected field [12]. Output of a line feed after: 80 chars 10 chars 32 chars 40 chars 64 chars } hardcopy mode only } reserved This automatic line feed (LF) can be combined with carriage return (CR) in hardcopy mode: entry in character 2 (PBH), bit 5=1.
6	RB1	Acknowledgment byte 1	X'20' - X'7F'	The sequence numbers entered in this character by the host are returned to it unmodified in the event of an acknowledgment being requested.
7	RB2	Acknowledgment byte 2	X'20' - X'7F'	
8	—	—	X'00'	reserved

Table 5-2: Description of the characters in PAR 10L/D - 70L/D

5

Incorrect Parameter Entries

The following errors are indicated in the status indicator line of the display terminal:

Indicator	Cause of error
GAD ?	A device address which could not be interpreted has been entered in character 3 (GAD).
PAR-EING	A device instruction which could not be interpreted has been entered in character 4 (GBF).

Table 5-3: Incorrect parameter entries

These indicators can be reset with RS.

5.3.2 Acknowledgments from Local Printers

When operating with acknowledgments, the status byte of a negative acknowledgment (SOM = X'42') indicates why the printer is not ready. In the case of a positive acknowledgment (SOM = X'41'), the status byte has the value X'40'.

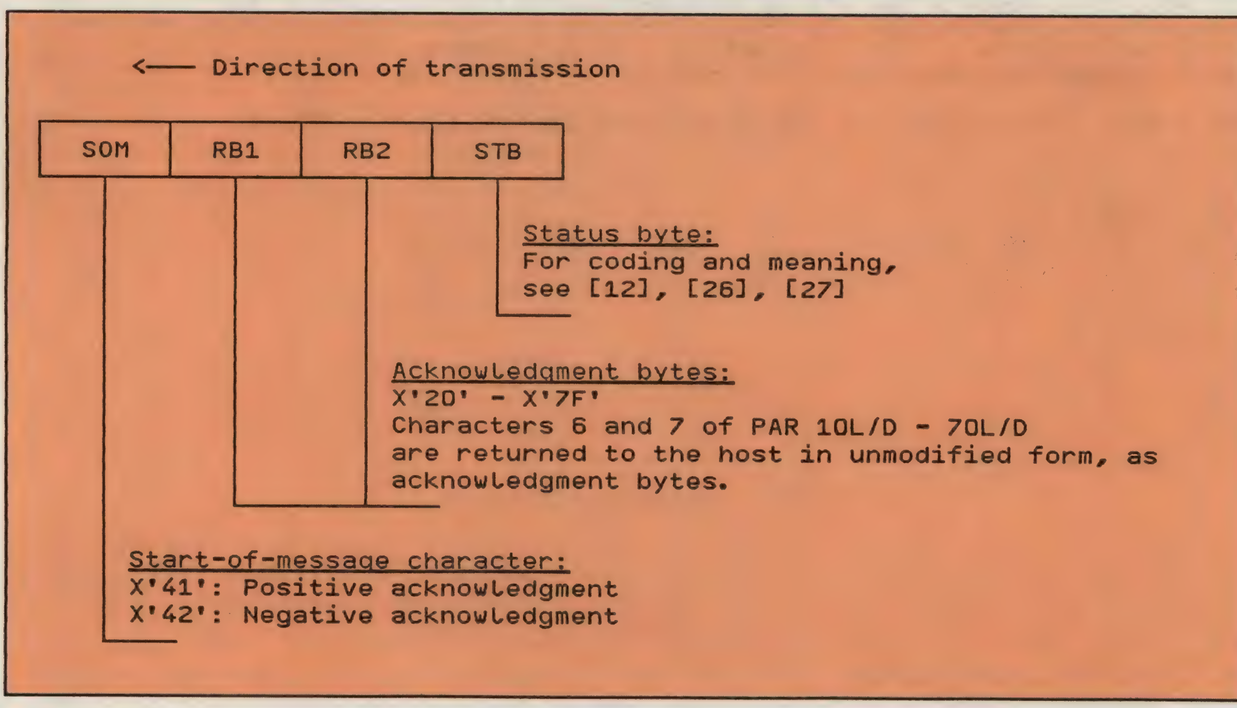


Fig. 5-5: Format of acknowledgments from local printers

5.3.3 Status Indicators for Local Printers

When a print operation is initiated from the data display terminal, any printer errors will be reported in the terminal's indicator line, in the form of status indicators.
Details concerning which of the errors listed in the table below can be indicated on which data display terminal are to be found in the relevant operating instructions.
The indicators can be reset with RS.

Indicator (DDT)	Coding		Meaning
	EBCDIC	ISO-7	
F0	4A	60	- Printer switched off - Printer not connected - There is no printer with this address
F1	C1	41	- Printer cover is open - Device error - Error in automatic single-sheet feed - Transmission error - Printer not connected - General printer error
F2	C2	42	- End of paper - End of ribbon - Ink reservoir empty
F4	C4	44	- Printer status HALT or STOP
F8	C8	48	- Format error (line too long) - Unknown control character transferred
FF	D6	4F	- Error in the printer message - Error at the printer interface - Adapter module missing or defective

Table 5-4: Status indicators for local printers

Note

When 9001 and 9004 Printers are used, only status indicators F0 and F1 are displayed.

5.4 DATA EXCHANGE WITH CENTRAL PRINTERS

In the case of central connection, a number of data display terminals can have access to one central printer.
A precondition for data exchange is that parameter ranges PAR 00D and PAR 10L/D - 70L/D be supplied with parameters.

The channel address (KAN) in PAR 10L/D - 70L/D represents the address of a printer terminal controller or a printer with an integrated printer controller connected to a local or remote cluster controller.
This channel address is identical for all data display terminals having access to the same printer.

5.4.1 Format of Parameter Ranges PAR 10L/D - 70L/D

The parameter ranges PAR 10L - 70L must be loaded if printing is to be initiated manually by pressing an LAx key on the data display terminal's keyboard.
The parameter ranges PAR 10D - 70D must be loaded if printing is to be initiated automatically via the host (characters 1/2, GEF1/2 of the PAR 00D = LA1 - LA2 or LA3 - LA7).
Parameter values can be supplied via the same message (data output), provided PAR 10D - 70D precedes PAR 00D in the message.
Assignment of the device functions to the corresponding parameter ranges is shown in Table 4-10.

Note

These parameter ranges need only be supplied with parameters if the user requires values other than those loaded as default values during bootstrapping.

Character	Designation	Meaning
1	KAN	Channel address
2	PBH	Parameter handling
3	GAD	Device address
4	GBF	Device instruction
5	ADA	Print command
6	RB1	Acknowledgment byte 1
7	RB2	Acknowledgment byte 2
8	——	reserved

Table 5-5: Summary of characters in PAR 10L/D - PAR 70L/D

Char.	Design.	Meaning	Coding	Explanation
1	KAN	Channel address	See tables 5-7 and 5-8	Addresses a printer terminal controller or an integrated printer controller.
2	PBH	Parameter handling	X'48'	PBH is not interpreted
3	GAD	Device address	X'00'	Reserved
4	GBF	Device instr.	X'53'	Data output (=write) only is possible via the printer.
5	ADA	Print command	Bit 7 6 5 4 3 2 1 Bit 1 1 (bit 7) 0 (bit 2)	=0: Hardcopy mode =1: Forms mode
			0 0 0 0 0 1 0 1 0 0 1 1 1 0 0 1 0 1 1 1 0 1 1 1	Output of a line feed after: 80 chars (hardcopy mode only) 10 chars (hardcopy mode only) 32 chars (hardcopy mode only) 40 chars (hardcopy mode only) 64 chars (hardcopy mode only) reserved reserved reserved
			Bit 6	=0: No acknowledgment requested after printout on printer connected to the printer terminal controller. =1: Acknowledgment requested after successful printing or termination of the print operation. The acknowledgment is passed to the host via the data display terminal.
6	RB1	Acknowledgment byte 1	X'20' - X'7F'	The sequence numbers entered in this character by the host are returned to the system unmodified if an acknowledgment is requested.
7	RB2	Acknowledgment byte 2	X'20' - X'7F'	
8	—	—	X'00'	Reserved

Table 5-6: Description of the characters in PAR 10L/D - 70L/D

Channel Address Values (Connection via 8112 Printer Terminal Controller)

Printer 1 address	EBCDIC	40	7F	5B	50	4D	5C	6B	4B	F0	F2	F4	F6	F8	7A	4C	6E	5A	7B	6C	7D	5D	4E	60	61	F1	F3	F5	F7	F9	5E	7E	6F
	ISO-7	20	22	24	26	28	2A	2C	2E	30	32	34	36	38	3A	3C	3E	21	23	25	27	29	2B	2D	2F	31	33	35	37	39	3B	3D	3F
Printer 2 address	EBCDIC	7C	C2	C4	C6	C8	D1	D3	D5	D7	D9	E3	E5	E7	E9	8C	6A	C1	C3	C5	C7	C9	D2	D4	D6	D8	E2	E4	E6	E8	BB	BD	6D
	ISO-7	40	42	44	46	48	4A	4C	4E	50	52	54	56	58	5A	5C	5E	41	43	45	47	49	4B	4D	4F	51	53	55	57	59	5B	5D	5F
Free printer address	EBCDIC	4A	82	84	86	88	91	93	95	97	99	A3	A5	A7	A9	4F	FF	81	83	85	87	89	92	94	96	98	A2	A4	A6	A8	FB	FD	07
	ISO-7	60	62	64	66	68	6A	6C	6E	70	72	74	76	78	7A	7C	7E	61	63	65	67	69	6B	6D	6F	71	73	75	77	79	7B	7D	7F
Channel		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32

Table 5-7: Channel address values (connection via 8112 Printer Terminal Controller)

Channel Address Values (Printer with 90037-103 Printer Controller)

Address	EBCDIC	40	7F	5B	50	4D	5C	6B	4B	F0	F2	F4	F6	F8	7A	4C	6E	5A	7B	6C	7D	5D	4E	60	61	F1	F3	F5	F7	F9	5E	7E	6F
	ISO-7	20	22	24	26	28	2A	2C	2E	30	32	34	36	38	3A	3C	3E	21	23	25	27	29	2B	2D	2F	31	33	35	37	39	3B	3D	3F
Channel		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32

Table 5-8: Channel address values (printer with 90037-103 Printer Controller)

5.4.2 Acknowledgments from Central Printers

When operating with acknowledgments, the status byte indicates a negative acknowledgment (SOM = X'42') indicates why the message could not be printed.

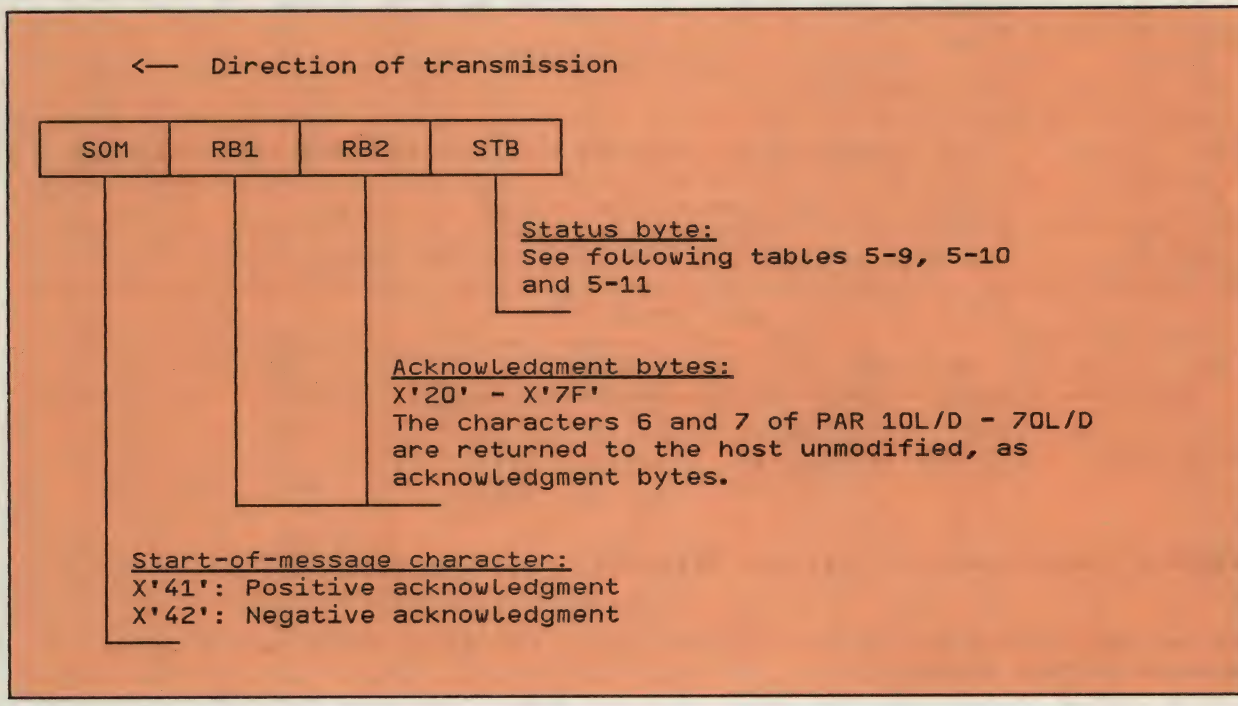


Fig. 5-6: Format of acknowledgments from central printers

Positive Acknowledgment

The message identified by RB1 and RB2 has been printed out normally.

Coding EBCDIC	Coding ISO-7	Meaning
7C	40	No indicators
C2	42	End of paper, but message has been printed with data key
C4	44	Data halt pressed before end of printing
C8	48	Format error (line too long) in the message
D1	4A	End of paper and format error
D3	4C	Data halt and format error

Table 5-9: Error messages

Negative Acknowledgment (without Printout)

The message identified by RB1 and RB2 could not be printed, and must be repeated by the system.

Coding EBCDIC	Coding ISO-7	Meaning
C1	41	- Device error - Cover open
C2	42	End of paper
C4	44	Data halt
D7	50	Data overflow; message was longer than the value set in the printer controller
4A	60	- Printer carriage cover open - No response from printer

Table 5-10: Negative acknowledgment (without printout)

Negative Acknowledgment (with Printout)

Printing of the message identified by RB1 and RB2 has been interrupted. The message is not lost, and can be printed out in full after the error has been eliminated.

Coding		Meaning
EBCDIC	ISO-7	
C6	46	End of paper
84	64	Printer carriage cover open

Table 5-11: Negative acknowledgment (with printout)

5

6 SOFTWARE COMPONENTS IN THE COMMUNICATION COMPUTER

The essentials of the TRANSDATA concept enable decentralized and distributed processing in computer networks.

These networks are realized by means of different communication computers of the 960 System.

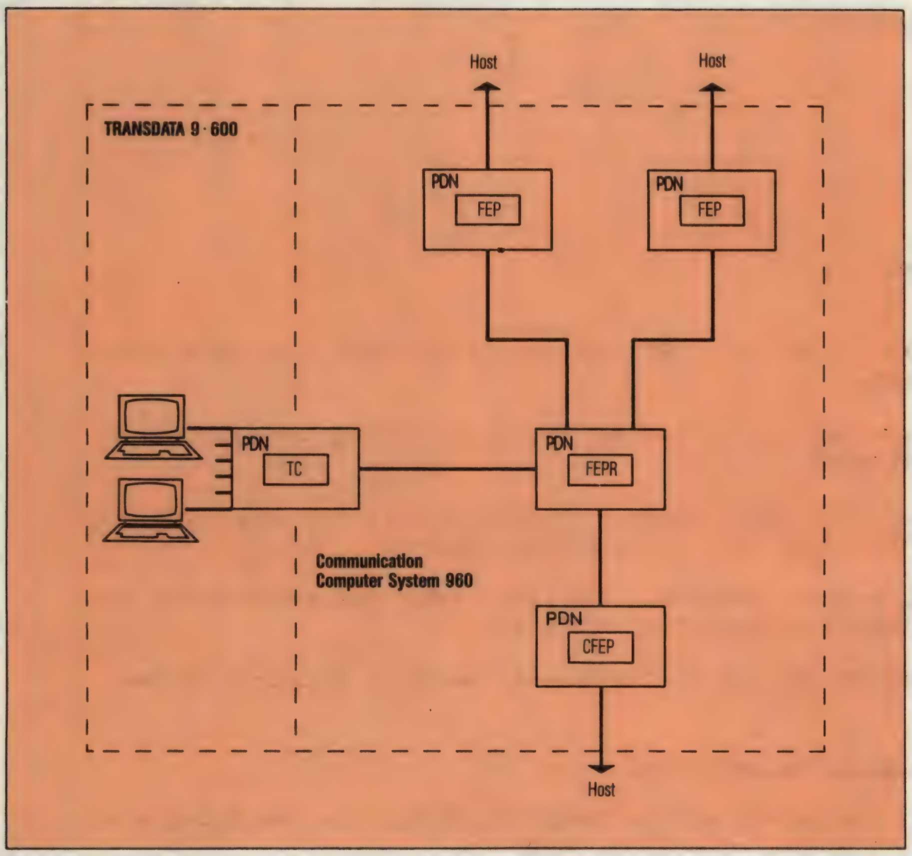


Fig. 6-1 Greatly simplified computer network

Communication tasks are performed by:

- (standard) front-end processors (FEP),
- remote front-end processors (FEPR) and
- compact front-end processors (CFEP).

The terminal computer realizes application-specific tasks in the sense of distributed processing.
The 960 Communication Computer System is controlled by the TRANSDATA PDN operating system.

6

The application programs in PDN (Program System for Teleprocessing and Network Control) can be formulated in the APS and COBOL programming languages.

The programmer has three possibilities available by which communication between the application program and the terminal is realized:

- MULTIKOS
  (Multi-Component Terminal Support),

- BERMUDA
  (User Service for Terminal Mask Support, see section 6.2 and [12] [15]),

- physical terminal support.

6.1 MULTIKOS

6.1.1 Brief Description

The programs formulated in APS or COBOL by the programmer can be executed in the terminal computer.

The application programs are supported by MULTIKOS during communication with data display terminals.

In the case of messages from the terminal (input side), the MULTIKOS provide a simple interface for the message processing.

In the case of outputs to the terminal, MULTIKOS help the programmer by editing messages intended for specific terminals.

What advantages does MULTIKOS (Multi-Component Terminal Support) offer the programmer?

<u>On the input side (input from terminal)</u>

The programmer can, in the applications program, forgo the handling and interpretation of device-specific message headers.

<u>On the output side (output to the terminal)</u>

The data display terminal, its components and device functions can be addressed by the programmer by means of uniform instructions on the logical level
(see "MULTIKOS Output Support").

As well as supporting message header and control character handling, MULTIKOS enables input and output coordination in the APS program e.g. assignment of acknowledgments to outputs to the printer [8],[12].

Integration into the APS Programming Language

The MULTIKOS functions available to the programmer at the APS language interface in the form of declarations and instructions.

Logical programming presupposes knowledge of the APS basic functions.

The following manuals contain information concerning the APS basic functions:

- APS manual [9]
- Interfaces to Data Communication [8].

Data display terminal functions which cannot be supported logically must be programmed physically (see chapter 4 and the section "Physical Support of Data Display Terminals").

Integration into the COBOL9600 Programming Language

The MULTIKOS functions are available to the programmer at the COBOL language interface in the form of procedure calls.

The COBOL basic functions are described in the "COBOL9600" manual.

The procedure calls

- CALL "SDEV @@ 01",
- CALL "SFUN @@ 01"

as well as their parameters and functions, for MULTIKOS support in COBOL programs, are described in the appendix of the "COBOL9600" manual.

The difference between the integration of MULTIKOS in COBOL9600 and in APS consists only in the different outward appearance of language.

The terminal support functions are identical in both cases.

6.1.2 Input Matching by means of MULTIKOS

The input matching function of the MULTIKOS Multi-Component Terminal Support provides the programmer with a terminal-independent message interface.

Messages from the data display terminal are, for example:

- key entries, DÜ1, DÜ2 and F1 through F5,
- short messages K1 through K14,
- inputs via the badge reader,
- printer acknowledgments.

Further information [8].

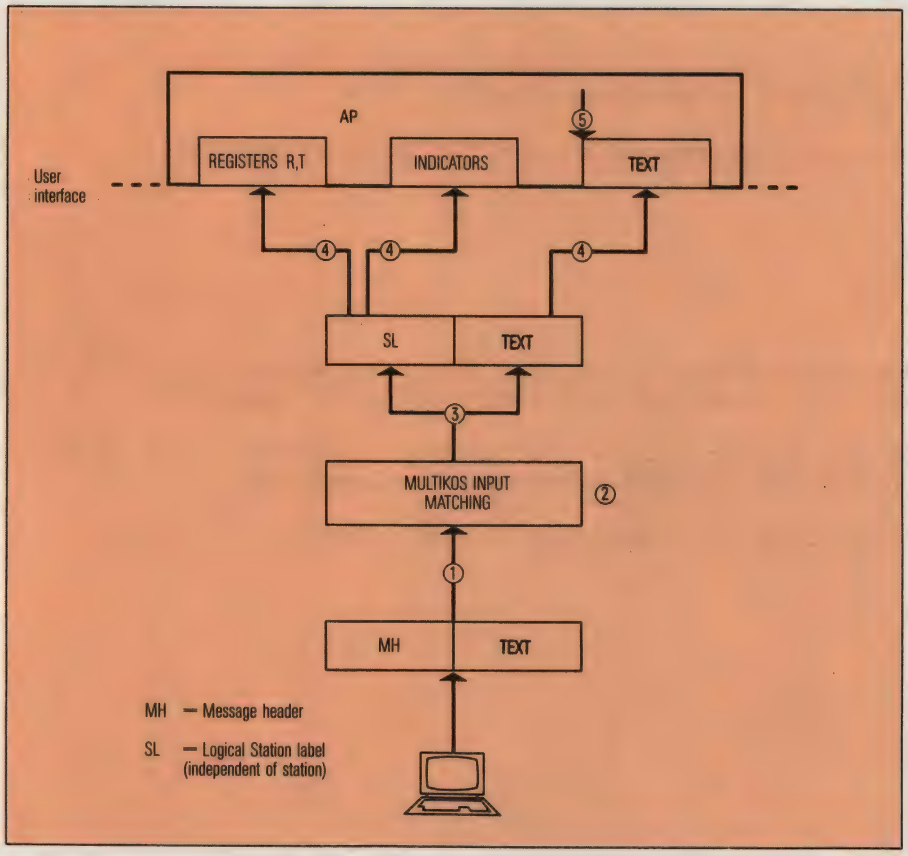


Fig. 6-2 Simplified presentation of message flow from the data display terminal to the application program

(1) The message from the display terminal is passed on to the MULTIKOS input matching function.

(2) The input matching function provides the application programs which are formulated in APS and COBOL with a terminal-independent input interface.

The input matching function converts, for this purpose, the message header (NK) to a logical station header (SE).

(3) The message that has been converted in this way (station header SE and text) is passed on to the application program.

(4) At the start of the application program, the registers and indicators are set by means of the station header.

(5) At the start of the application program, the input buffer pointer points to the first character of text information from the data display terminal.

Registers

Register R contains information concerning the progress of a preceding output job (e.g. output to printer [12]).

Information examples

- Output job terminated positively
- Error taken place (e.g. paper out at printer)
- Receiver of output job cannot be reached (e.g. timeout, address error)
- Error message.

Register T contains additional information concerning the message from the terminal.

Examples of additional information

- Code of the corresponding short message
- Code of the corresponding transmission key
- Badge withdrawal code
- Code of the blocks subsequent to a blocked message
- Error message code.

Indicators

MULTIKOS provides the user with indicators at the start of the application program. These indicators enable him to obtain additional information about the current status of the terminal or about the current message.

Examples of indicators

- Keyboard operable/inoperable (Indicator *110)
- Printer operable/inoperable (Indicator *107)
- Badge in reader/not in reader (Indicator *097)
- Key status (Indicators *100 - *102)
- Message begins with positioning string/with text (Indicator *104).

The indicators and registers can be interrogated by means of the APS instruction 'YIF' [9].

Further information concerning indicators and registers [8].

Error messages

In the following cases, MULTIKOS passes an error message to the application program instead of the message from the terminal.

Examples of error messages

- Message header cannot be interpreted (NBZ invalid)
- Field mode specified without BERMUDA
- Error in MULTIKOS input matching.

Further information concerning error messages [8].

6.1.3 MULTIKOS Output Support

MULTIKOS output support aids the programmer in the formatting of an output message destined for the display terminal.

MULTIKOS assembles, for this purpose, declarations for generating device lists and instructions for message construction.

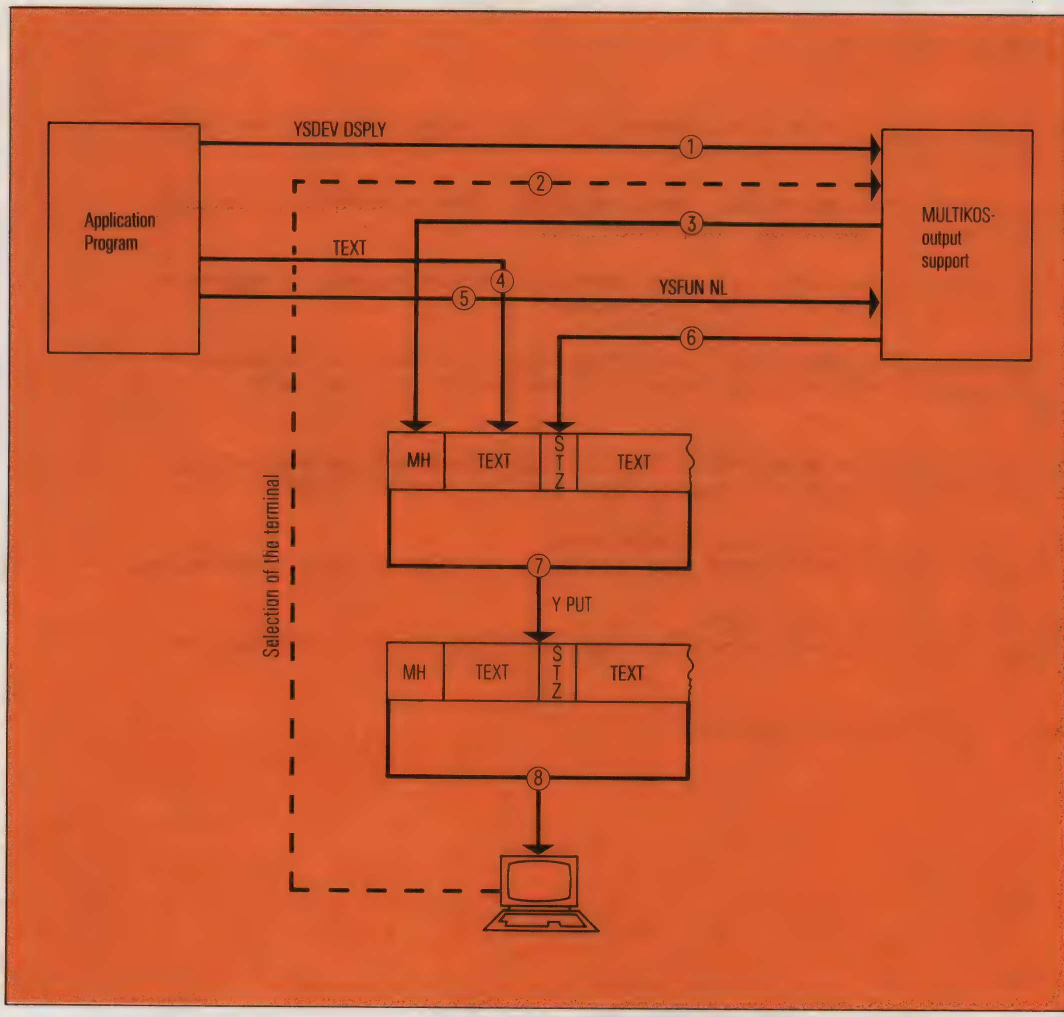
The programmer addresses the data display terminal, its output components and device functions by means of symbolic names.

This procedure obviates the need for the programmer to have knowledge of the physical construction of message headers, parameter ranges and device functions.

The user has two instructions available for the purpose of constructing message headers and device control characters in the output buffer.

- YSDEV to select a terminal output component.
  This instruction constructs the message header at the same time.
- YSFUN to select a device function.
  This instruction enters the physical device control character in the message.

See also the examples of output support.



STZ = Physical device control character

Fig. 6-3 Simplified presentation of data display support

(1) Calling of instruction YSDEV in the application program. The instruction is supplied with the symbolic name of the desired component e.g. DSPLY assigned to the screen.
(2) MULTIKOS finds out by means of internal tables the terminal type, e.g. 9750.
(3) The data assigned to the terminal DSPLY is entered in the message as the message header.
(4) The user moves his text into the output buffer.
(5) Calling the YSFUN instruction in the application program. The instruction is supplied with the symbolic name of the desired device function, e.g. 'NL' for new line.
(6) The symbolic name of the device function is converted to a control character specific to the device and is entered in the message.
(7) The constructed message is sent from the application program to the data display terminal using the instruction 'YPUT'.
(8) The message (message header, text and possible control characters) is interpreted by the display terminal.

MULTIKOS has a precise image of the existing output components (e.g. screen, printer, badge reader) and of the possible device functions for each component.

In MULTIKOS, each output component is assigned a certain message header, the so-called standard message header.

In certain cases (see example), the application programmer can modify or extend this message header.

For this purpose, the following declarations are available:

- YDDEV enables definition of a device table and
- YDDL enables definition of a device list.

See also chapter "Function calls for modifying the standard message header".

Example

In an application program for the 9750, outputs are displayed on the screen and outputted at a local printer.

The 9752 Data Display Terminal has, on the other hand, no local printer.

In order that the application program can be used, nevertheless, for both terminals, a device list is defined for the 9752 terminal using the YDDEV instruction:

```
YDDEV   9752,(PRNT,DSPLY)
```

If the application program is connected to a 9752 terminal, the message header for the screen in entered in the output buffer instead of the message header for the printer.

The outputs to the printer are then diverted automatically to the screen.

The declaration YDDL is necessary if such modifications are to be carried out for several device types.

Further information [8].

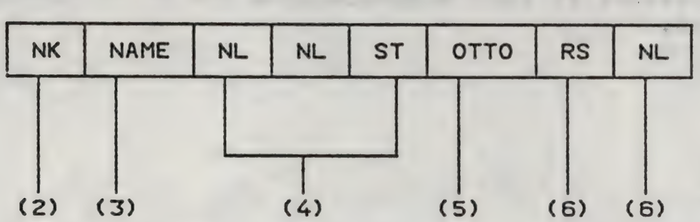
EXAMPLE OF OUTPUT SUPPORT

Example 1: Without device list declaration

(1)	YGTB	AUS	Output buffer is activated.
(2)	YSDEV	DSPLY	A standard message header is placed in the output buffer.
(3)	YMVC	-,AUS,TEXT	The content (content is 'NAME') of the source field with the symbolic address 'TEXT' is entered in the output buffer.
(4)	YSFUN	(NL,2),ST ——>	By means of this control character, the following text is displayed, flashing.
		——>	By means of this control character, the cursor is positioned two lines forwards and is set at the start of this line.
(5)	YMVC	-,AUS,'OTTO'	The character string 'OTTO' is entered as message in the output buffer.
(6)	YSFUN	RS,NL	
		——>	'Blinken' (Flashing) is reset.
		——>	The cursor is positioned one line forwards. The device-specific control character is entered in the output buffer.
(7)	YPUT	Y	The output buffer contents (message) are sent to the locally on nected data display terminal.

Content of the output buffer before the YPUT call

(1) AUS



Example 2: With device tables

Declaration section of the application program

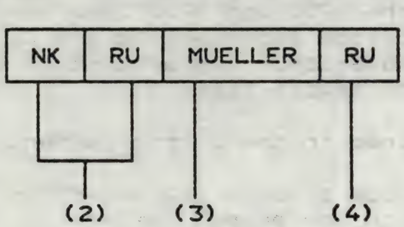
DEV1	YDDEV	9750,(DSPLY,(RM),(RU))	YDDEV enables the addressing of data display terminals, their components (DSPLY) and device functions (RM, RU). DEV1 is the symbolic address of the device list.

Instruction section of the APS program

(1)	YGTB	AUS	
(2)	YSDEV	DSPLY,DEV1	The component DSPLY of the display terminal is selected. DEV1 is the symbolic address of the device list (YDDEV call).
(3)	YMVC	-,AUS,TEXT	The contents of the source field with the address 'TEXT' (content 'MUELLER') is entered in the output buffer.
(4)	YSFUN	RU	The physical device character for 'RU' is entered in the output buffer.
(5)	YPUT	Y	The content (message) of the output buffer is sent to the locally connected data display terminal.

Content of the output buffer before the YPUT call

(1) AUS



If, in the example mentioned above, the locally connected data display terminal is a 9750, it receives a message header with the send instruction "LESE MODIFIZIERTE FELDER ohne NUL" (RM). ('read modified fields without NUL's)

The ROLL-UP function is initiated before the first text character is displayed.

If the data display terminal is not a 9750, the terminal is provided with an output buffer having the standard message header for the screen.

Example 3: With a device-list declaration

DEV1	YDDL		Device list is initiated. DEV1 is the symbolic name of the device list.
	YDDEV	9750,(DSPLY,(NL),(CL))	In the case of the 9750 Data Display Terminal, the standard message header of the component DSPLY is extended by the addition of the device functions NL and CL.
	YDDEV	9752,(DSPLY,(NL),(CK)), (PRNT,DSPLY)	The standard message header of the component DSPLY, in the case of the 9752 Data Display Terminal, is extended by the addition of the device functions NL and CL. The standard message header for the PRNT component is replaced by the message header for the DSPLY component.
	YDDL	END	Termination of the device list.

Instruction section of the APS program

(1)	YGTB	AUS	Activate buffer.
(2)	YSDEV	DSPLY,DEV1	The component DSPLY of the locally connected terminal is selected and the message header is entered in the output buffer. DEV1 is the symbolic address of the device list. In the case of the 9750 and 9752 Data Display Terminals, the device functions NL and CL are also entered automatically on account of the definition in the device list.
(3)	YMVC	-,AUS,'MUELLER'	The text is entered in the output buffer.
(4)	YSFUN	NL	The physical device control character for 'NL' is entered in the output buffer.
	YPUT	Y	The output buffer contents are sent to the locally connected terminal.

Output buffer contents for the 9750 and 9752 Data Display Terminals before the YPUT call

(1)	AUS	NK for screen	NL	CL	MUELLER	NL
		(2)	(2)	(2)	(3)	(4)

(5)	YGTB	AUS	Activate output buffer.
(6)	YSDEV	PRNT,DEV1	The PRNT component is selected and the message header entered in the output buffer. In the case of the 9750 Data Display Terminal, the standard message header for the PRNT component is entered. In the case of the 9752 Data Display Terminal however, the message header for the screen is entered, on account of the device list.
(7)	YMVC	-,AUS,'MUELLER'	The text 'MUELLER' is entered in the output buffer.
(8)	YSFUN	NL	The physical device control character for 'NL' is entered in the output buffer.
	YPUT	Y	The contents of the output buffer are sent to the locally connected display terminal.

Content of the output buffer before the YPUT call

For 9750:

(5)	Aus	NK for printer	MUELLER	NL
		(6)	(7)	(8)

For 9752:

(5)	Aus	NK for screen	NL	CL	MUELLER	NL
		(6)	(6)	(6)	(7)	(8)

6.1.4 MULTIKOS for the 9750, 9752 Data Display Terminals

Supported data display terminals and extras

The following table shows the versions of those devices supported by MULTIKOS.

Data display terminal	Support for both input and output in logical mode
9750-1 9750-2 9750-3 9750-4 9750-5 9750-7 9750-8 9750-9	From PDN version 6.0
9752-1 9752-2	From PDN version 8.0 Since the 9752 Data Display Terminal is compatible with the 9750 Data Display Terminal, support is already possible from PDN V6.0. <u>Prerequisite:</u> The 9752 Data Display Terminal must be generated as the 9750 DD Terminal.
9750-62	From PDN version 6.0 To use the special text editing functions, the DDT 9750-62 must be generated as the 8162 DDT.

Table 6-1 Versions of the 9750 and 9752 Data Display Terminals that can be supported logically by MULTIKOS.

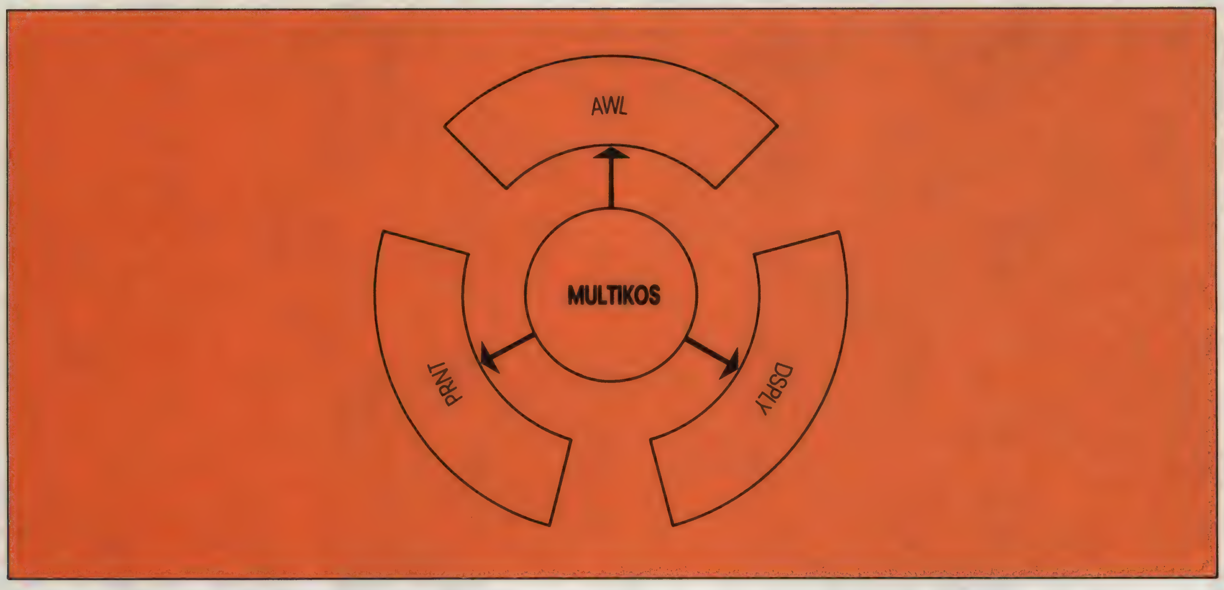
Supported operating modes

MULTIKOS supports applications in the record mode in the case of 9750 and 9752 Data Display Terminals. The use of the field mode is possible only with the 9750 DDT in conjunction with BERMUDA.

Output components

Every display terminal output component in the MULTIKOS virtual multi-component terminal is addressed by the application program by means of a symbolic name.

6



Symbolic name	Terminal component
DSPLY	Screen (display)
PRNT	Printer
AWL	Automatic badge reader

Fig. 6-4 Output components of the 9750 and 9752 Data Display Terminals with regard to MULTIKOS.

System line

The system line is not supported by MULTIKOS.

Further Information [8].

6.1.4.1 Screen Support (DSPLY Component)

If no device-list is specified in the YSDEV instruction, a standard message header is placed in the message.

This so-called standard message header causes specific functions for the screen of the 9750 and 9752 Data Display Terminals (see following paragraph).

Standard message header in the case of DSPLY component specification

Contained in the following table are, on the left, the functions of the standard message header and, on the right, the corresponding specifications in the case of physical programming.

This table provides the programmer with an overview of those physical functions of terminals that are supported logically by MULTIKOS.

Standard message header	Physical programming
Transmit instruction: Send unprotected fields starting from cursor. When a transmission is initiated by the user, using a DÜ key, the data in all unprotected fields from the cursor position to the next end marker or to the end of the screen is transmitted.	X'42' in character SAW1 of PAR OOL.
Insert line (EFZ) and delete line (AFZ). EFZ and AFZ are permitted via the keyboard.	Bit 1 in character FST1 of PAR OOL=0.
On input of EFZ, all field delimiters and all data from the last line are copied to the cursor line.	Bit 2 in character FST1 of PAR OOL=1.
On input of AFZ, all field delimiters and all data from the cursor line are copied into the last line.	Bit 3 in character FST1 of PAR OOL=1.
The 'RU' function is initiated via the keyboard	Bit 5 in character FST1 of PAR OOL=0.
The 'RUB' mode is deactivated.	Bit 6 in character FST1 of PAR OOL=0.
The automatic tabulator function is on. The user cannot move the cursor to protected fields.	Bit 3 in character FST2 of PAR OOL=1.
'Clear memory' (LSP) permitted via keyboard.	Bit 1 in character FST2 of PAR OOL=0.
The cursor tracks the data output. On completion of the output, the cursor is psitioned behind the last outputted character.	Bit 1 in character FST3 of PAR OOL=0.
In the case of data input, the positioning instructions are specified according to version B (see chapter 4).	Bit 2 in character FST3 of PAR OOL=0.
After completion of data input, the cursor is positioned after the effective end marker. If no end marker is present, the cursor is situated at the start of screen.	Bit 4 in character FST3 of PAR OOL=0.
The keyboard is released immediately after data input. Dialog mode off.	Bit 5 in character FST3 of PAR OOL=0.

Table 6-2 Standard message header in the case of DSPLY component specification

Function calls for modifying the standard message header

The standard message header created by the YSDEV DSPLY call can also, with regard to some functions, be modified subsequently (see also MULTIKOS Output Support).

This modification is undertaken by the appropriate YSFUN calls.

The following table contains function calls and the corresponding responses.

YSFUN	Response	Physical programming
DI	Dialog mode on. After data input the keyboard is locked until the next output.	Bit 5 in character FST3 of PAR OOL is set to 1.
CK	Keyboard is locked immediately on output of data. <u>Note</u> The keyboard remains locked until the application program sends an output to the terminal without 'CK'. Short messages to the host are always possible.	Bit 6 in character FST3 of PAR OOL is set to 1.
RM	At the next input, the field contents (without NULs) and the start address of field are sent for each modified field. In the case of the 9750-62 Data Display Terminal, additional fields, defined as protected and readable (by PR), are transmitted even if they have not been modified.	Character SAW1 in PAR OOL is set to X'45'.

Table 6-3 Function calls for modifying the standard message header

If functions are required that, owing to 'default' in the message header or the three possibilities for modification are not realizable, then the terminal must be programmed in the physical mode (see chapter 'Handling of non-supported functions').

Function calls for the DSPLY component

The device functions for controlling the terminal screen can be called up using the MULTIKOS instruction YSFUN.

The functions are divided into five groups.

In the following tables, the logical functions and the response to them within the terminal are compared with the device functions (see chapter 4).

Group 1: Delete functions

Format of call: YSFUN <fun>

Device function (see chap.4)	<fun>	Response within the data display terminal.
LVD	CF	All characters of an unprotected field are cleared from the cursor position up to and including any end marker, or up to the end of screen. The cursor position remains unchanged.
LZF	CL	All characters of an unprotected field are cleared from the cursor position to the end of the line or field, or to an end marker. The cursor remains in its original position.
LSP	NP	Clearance of the entire screen using NULs. The cursor is positioned subsequently at the start of screen. The operand area or message header is not cleared.

Table 6-4 Delete functions

Group 2: Positioning specifications

Format of call: YSFUN <fun>,<spec>

Device function (see chap.4)	<fun>	Meaning <spec>	Response within the data display terminal
IS4 sequence	LI	Line number	Cursor is set to the start of the specified line, in column 1. When followed by CO, the control character string becomes a complete positioning instruction with both line and column specifications.
IS4 sequence	CO	Column number	Cursor is positioned to the specified column of a line previously defined using LI. If a preceding LI specification is omitted, the cursor is positioned in the specified column of the first line.
SNZ	NL	Number of lines	The cursor is positioned forwards the number of lines specified in <spec> and placed at the start of the line. The cursor skips from the 24th line to line 1. <u>With regard to the 9750-62:</u> When line 24 is reached, the screen format is shifted upwards by one line. The cursor skips to line 25. When line 125 is reached an audible alarm is heard. When line 125 is reached, the cursor stops and the audible alarm sounds.
RU	RU	Number of line feeds	The characters in lines 2 to 24 are shifted up by one line. The characters in line 1 are lost. In the unprotected 24th line, NUL characters are displayed in the same way as at the end of the 23rd line. The cursor is positioned at the begin of line 24. If <spec> is specified, the function is repeated accordingly.
	DC	Number of NUL chars.	NUL characters are outputted, starting at the cursor, in accordance with the number specified in <spec>.

Table 6-5 Positioning specifications

Note on <spec>

If <spec> is not specified, 1 is assumed automatically.

Note on the RU specification

Using the RU function for screen enables line 24 to be defined beforehand as "unprotected" by means of a field handling character.

The function code 'RU' needs now to be followed only by a function call for text display or for defining a protected field.

The specification of 'RS' following 'RU' is not allowed.

Group 3: Text display at the 9750 and 9752 Terminals

By means of these function codes, video attributes can be specified.

A function code for text display is effective on the screen from its input position to the position where a new function code for text display follows.

The following three display criteria can be selected in any combination whatever beginning from the PDN version 8.0.

- Character font
  - standard character font
  - reverse or underscored
- Brightness (intensity)
  - bright
  - normal intensity
- Illumination
  - flashing
  - non-flashing.

In the case of the 9752 Data Display Terminal, the criteria character font and brightness (intensity) contribute to the display in colour.

6

Text display on the 9750 Data Display Terminal

Call format: YSFUN <fun>

Physical display control char. (see chap. 4)	<fun>	Response within the data display terminal
NI, SF, NF	NH BR	The following text section is displayed bright with standard character font.
NI, SF, FL	NHB ST	The following text section is displayed bright, flashing and with standard character font.
NI, SF, FL	NN ME	The following text section is displayed moderately bright with standard character font.
RI, SF, FL	NNB	The following text section is displayed moderately bright, flashing and with standard character font.
NI, US, NF	IH IT	The following text section is displayed bright with text underscored or reverse (for setting in the data display terminal, see chapter 4).
NI, US, FL	IHB	The following text is displayed bright, flashing with text underscored or reverse (for setting in the terminal, see chapter 4).
RI, US, NF	IN	The following text section is displayed moderately bright with the text reverse or underscored (for setting in the terminal, see chapter 4).
RI, US, FL	INB	The following text section is displayed moderately bright, flashing with the text reverse or underscored (for setting in the terminal, see chapter 4).
BL	DK	The following screen section contains blanked out characters (display is suppressed).

Table 6-6 Text display for 9750

Text display on the 9752 Data Display Terminal

The 9752 Data Display Terminal, beginning from the PDN version 8.0, is supported logically by MULTIKOS when fields are displayed in green, yellow, red and white.

The function for flashing can be additionally activated for each field.

Call format: YSFUN <fun>

Physical display control char. (see chap. 4)	<fun>	Response within the data display terminal
NI, SF, NF	GR NH BR	The following text section is displayed green with standard character font.
NI, SF, FL	GRB NHB BR	The following text section is displayed green, flashing with standard character font.
RI, SF, NF	GE NE ME	The following text section is displayed yellow with standard character font.
RI, SF, FL	GEB NNB	The following text section is displayed yellow, flashing with standard character font.
NI, US, NF	RO IH IT	The following text section is displayed red with standard character font.
NI, US, FL	ROB IHB	The following text section is displayed red, flashing with standard character font.
RI, US, NF	WE IN	The following text section is displayed white with standard character font.
RI, US, FL	WEB INB	The following text section is displayed white, flashing with standard character font.
BL	DK	The following screen section contains blanked out characters (display is suppressed).

Table 6-7 Text display for the 9752

Note

Programs with function calls for field display in the 9750 (see alternative specifications under <fun>) can also be executed using the 9752 Display Terminal without program modification even though these calls existed before the PDN version 8.0.

Group 4: Field handling

Call format: YSFUN <fun>

A function code for field handling is effective on the screen from its input position to the position where a new function code for field handling follows.

For use of field handling characters that are not supported by MULTIKOS, see chapter 'Handling uf unsupported functions'.

Physical field handling char. (see chap. 4)	<fun>	Response within the data display terminal
P, NM, PF, A	PT	The following screen section is defined as protected, numeric, alphanumeric, printable and not readable.
PS, NM, PF, A	PR	The following screen section is defined as protected, numeric, alphanumeric, printable and readable.
V, NM, PF, A	UP	The following screen section is defined as unprotected, variable, numeric, alphanumeric, printable and readable.
V, NM, PF, A	NU	The following screen section is defined as numeric, alphanumeric, variable, printable, unprotected and readable.

Table 6-8 Field handling

Group 5: Special Functions

Resetting text presentation and field handling

Call format: YSFUN <fun>

Physical programming (see chap. 4)	<fun>	Response within the data display terminal
Text display: IS3 sequence Field handling: IS2 sequence	RS	The following screen section is defined as unprotected, not numeric and is displayed bright, non-flashing and with standard character font. In the 9752: bright is displayed as green.

Table 6-9 Text display and field handling resetting

Note

The specification of 'RS' following 'RU' is not allowed.

Initiation of Hard Copy Function

Call format: YSFUN <fun>

Physical device function (see chap. 4)	<fun>	Printer response
LA1	HC	Produces hard copy of the screen contents from the cursor position to the end marker or to the end of screen. Output is to local printer. With 9750-62: All five screen memory contents are printed if no end marker is encountered.

Table 6-10 Hard copy of screen

Note

The hardcopy function is initiated only when data transmission is completed and the whole buffer has been processed by the terminal, meaning, in effect, that the useful information has been displayed on the screen and that all device functions, with the exception of RM, HC and DI, have been executed.

Before the output buffer is sent off, the cursor is positioned at the start of the text to be printed (using functions LI and CO) because the printing of the message proceeds from the cursor position.

In the case of the 9752 Data Display Terminal, specifying 'HC' has no effect.

6.1.4.2 Support for the Badge Reader (AWL Component)

Standard Message Header in case of Specification of AWL Component

- Request for badge input via badge reader (physical programming, see chapter 4, bit 1 in character FST8 of PAR 01L).
- Execution of a roll-up function on the terminal screen.
- The following prompt appears in line 24 on the screen:
  'BITTE AUSWEIS EINGEBEN!'
  (please insert badge)
- The keyboard is locked (exception: key sequence 'ESC:').

Note

For formatting a request for a badge input, only the call YSDEV AWL is necessary.

Badge Reader Support at the Data Display Terminal

The application program requests badge input by means of the call 'YSDEV' (see MULTIKOS output support with message header).

After badge request, it is to perform either one of those actions at the terminal which are mentioned below:

- Badge input whereby the badge contents are transmitted automatically to the terminal computer.

  The keyboard is released automatically after badge input.

  The application program is notified with each message from the terminal via the indicator *97 as long as a badge is still in the badge reader [8].

  The application program is notified of the removal of the badge in the form of the short message K14 in register T[8].

- Input of the key sequence 'ESC:' if badge cannot be inserted.

  On keyboards where the colon has to be entered via the shift key, this key is pressed before the ESC key in order to effect 'ESC'.

  After 'ESC:' has been inputted, the keyboard remains locked. The cancellation of a badge request must come from the application program.

Function call for the component AWL:

Format call: YSFUN RS

The request for badge input at the terminal is canceled.

A roll-up function is executed on the screen.

The keyboard is released.

Line 24 remains protected.

Note

The cancelation of the request for badge input by the application program should be displayed on the terminal screen in the form of a message.

For this purpose, the function calls for controlling the screen (text display and field handling) can be used.

6

6.1.4.3 Support for the Printer (PRNT Component)

The following table contains, on the left, the message header functions and, on the right, the corresponding specifications for physical programming.

Standard message header	Physical programming
The local printer at the device address 1 of the terminal is addressed.	Device function LA1 in character GEF1 of PAR 00D, Channel address X'00' in character KAN of PAR 10D.
Forms mode on, meaning that all print control characters must be contained within the text.	Bit 1 in character ADA of PAR 10D=1.
Output to printer does not affect the terminal screen. Bypass mode!	Character LAP1 of PAR 00D=X'41'.
Positive and negative acknowledgment is requested.	Bit 1=0 Bit 6=1 in character PBH of PAR 10D.
Acknowledgment bytes = X'7C7C' (EBCDIC). For structure of acknowledgments, see [8].	Character RB1/RB2 in PAR 10D.

Table 6-11 Standard message header in the case of the component PRNT

Note

MULTIKOS does not monitor the maximum line length of the printer. The function characters for line feed e.g. NL must be inserted by the programmer in the correct position within the text.

Further information concerning:

- component PRNT and its function calls [8],
- printer [12].

6.1.5 Cross-Terminal Traffic

Cross-terminal traffic enables screen contents to be printed on a central printer of the communication system.

Contrary to hardcopy output (function HC in the case of the component DSPLY) or output to a printer using 'YSDEV PRNT', the print out does not occur via the printer connected locally to the data display terminal.

The print out occurs on a central printer of the communication computer.

For the initiation of cross-terminal traffic, the parameter ranges PAR 10L (initiation by terminal user) or PAR 10D (initiation by the application program) must be supplied appropriately.

This input is handled by the procedure '$YMIN', with which the programmer is provided, from PDN Version 8.0 onwards.

The terminal user can execute input to the parameter ranges himself.

For this purpose he has the administration command 'INLA' [10].

The indirect initiation of cross-terminal traffic (initiation by the application program) has to be programmed physically.

Further information concerning cross-terminal traffic [8].

6.1.6 Handling of Unsupported Functions

The logical support of the 9750 and 9752 Data Display Terminals does not cover the entire spectrum of device functions that are possible.

If such functions are needed, one can proceed in the following way:

1) All device functions that are programmable by means of a two-character sequence or a three-character sequence in the text (see chapter 4) can be written, at all times, into the message text using the APS instruction 'YMVB'.

   Note

   The control character may only be specified in EBCDIC! The physical mode need not be switched on.

Examples of functions

AFG - Delete character

EFG - Insert character

AFZ - Delete line

EFZ - Insert line

SMR - Cursor right

SML - Cursor left

SMO - Cursor up

SMU - Cursor down

SZA - Cursor to start of line

SBA - Cursor to start of screen

SDZ - Cursor to start of line above

TAR - Tabulator right

TAL - Tabulator left

LVA - Erase variable data to specified address

FAZ - Set field separators to initial state

MAR - Mark field

AM - Start marker

EM - End marker

BEL - Visual alarm

BRS - Reset BEL

AKA - Audible alarm

SS - Set cursor

WDH - Repeat

LZE - Logical end of line

P - Programming P keys

2) The parameter range for the system line can be written, at all times, using the APS instruction YMVB, into the output buffer (EBCDIC code!).

3) When device functions are used that can only be entered in the parameter ranges, the terminal must be operated in the physical mode (see physical support for data display terminals).

Example of functions

RUB - ROLL-UP mode

LA functions (except LA1)

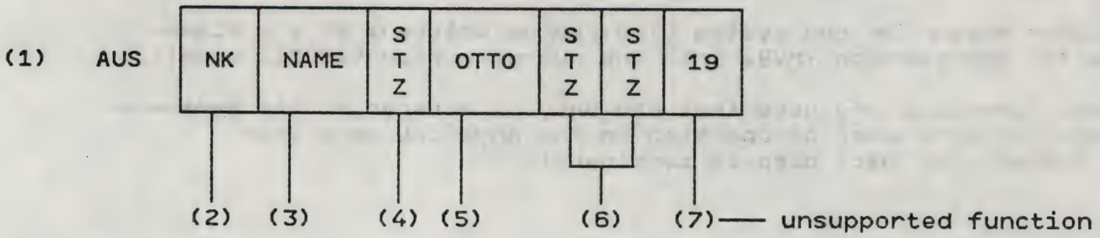
Modification of the parameter ranges e.g.

- special send instructions
- user inhibit for LA and P keys.

EXAMPLE OF PROGRAMMING UNSUPPORTED FUNCTIONS IN THE APS PROGRAM

(1)	YGTB AUS . . .	Activate output buffer
(2)	YSDEV DSPLY . . .	This specification causes a message header for the screen to be placed in the output buffer.
(3)	YMVC -,AUS,TEXT . . .	The content (content is 'NAME') of the source field with the address 'TEXT' is entered in the output buffer.
(4)	YSFUN NNB . . .	The following text is displayed moderately bright, flashing with standard character font.
(5)	YMVC -,AUS,'OTTO' . . .	The character string 'OTTO' is placed in the output buffer.
(6)	YSFUN RS,NL . .	NNB is reset. The cursor is then moved to the start of the next line.
(7)	YMVB -,AUS,'19' . . .	The end marker is set to the start of the new line.
(8)	YPUT Y	The contents of the output buffer are outputted to the logically connected terminal.

Content of the output buffer before the YPUT call:



6.1.7 Physical Support for Data Display Terminals

The physical mode for operating a data display terminal is the to be selected when the programmer wants to use terminal functions no supported by MULTIKOS (see 'Handling of unsupported functions').

The programmer is not supported by MULTIKOS either for input or output.

The physical mode requires from the programmer a precise knowledge of the terminals concerned as well as of the relevant message headers and control characters (see chapter 4).

With inputs from the terminal, the message is passed unchanged to the application program.

With outputs to the terminal, the programmer must construct the entire message himself by entering the message header and the physical control characters (see chapter 4).

Operation of the physical mode has, moreover, the following effect:

- Virtual programming via specification of symbolic names for output components and device functions is relinquished.

- Co-ordination of input and output is renunciated.

The physical mode is activated by the indicator *127 [8].

The indicator can already be pre-set by means of the operand PMODE = P during the generation of the XSTWA module [11].

The indicator can also though be set and reset dynamically in the application program:
YSON - Indicator is set

YSOFF - Indicator is reset [9]

Note

The physical mode is to be regarded as an alternative to logical support by MULTIKOS (YSDEV/YSFUN).

An application of both modes is not advisable.

6.1.8 Rules for Generation (MULTIKOS)

The linkage of MULTIKOS is controlled by the STAS operand of the macro XSTAT in the configuration source program [10].

- The specification of

  STAS=MULTIKOS or BERMUDA, BERMUDA/P, BERMUDA/R, BERMUDA/PR

  causes both the input matching and output support section to be linked.

  The input matching function is placed in the input sequence of the terminal.

- Should a local printer be supported at the 9750 Data Display Terminal (HC function or PRNT component), the operand STATTYP of the XSTAT macro is to be specified with 'DSS-9750+HC'.

  In the case of the 9752 Data Display Terminal, the supplement '+HC' is omitted (no local printer).

- Should the badge reader be operated at the 9750 and 9752 Data Display Terminals, the connection of the badge reader is to be made known in the following manner:

  The operand OPCH=<name> is to be specified in the XSTAT macro for the 9750 and 9752 Data Display Terminals. The operand AWL = JA must be specified in the relevant XOPCH macro.

- Where inputs in sequences defined by STAS = USERx are to supported by the MULTIKOS input matching function, the module YMEIN must be placed as the first module in the appropriate input sequence.

  Further information concerning generation [10], [11].

6.2 BERMUDA

6.2.1 Brief Description

BERMUDA (User Service for Terminal Mask Support) is a component of the PDN operating system for use in the Siemens System TRANSDATA 9.600 which is composed of terminal computers and data display terminals. BERMUDA is used by data display terminals that are connected to terminal computers employed by the TRANSDATA 960 System.

BERMUDA is employed for the following tasks:

- input support for formatted data
- formatted data output
- editing formatted data for printout
- relieving the host computer of
  - terminal control
  - terminal user support
  - data validation functions
  - data storage.

All other TRANSDATA PDN facilities can be used in addition to BERMUDA.

The 9750, 9752, 9770 and 8160 Data Display Terminals are supported in the PDN version 2.0.

This chapter provides an overview of the support given to the 9750 and 9752 Data Display Terminals with regard to terminal functions in association with BERMUDA.

Note

The 9752 Data Display Terminal must be generated as the 9750!

What are the advantages of BERMUDA with regard to programming?

The programmer does not need to know the physical attributes of the data display terminal.

This means that the programmer does not have to handle or interprete device-specific message headers on the input side.

On the output side, BERMUDA obviates the need for the programmer to have knowledge of the physical construction of message headers, parameter ranges and device functions.

This support, which takes place in the background, cannot be influenced by the programmer.

The programming of an application program presupposes knowledge of COBOL, APS and PDN.

Further information concerning BERMUDA [13].

6.2.2 Use of BERMUDA with the 9750 Data Display Terminal

Operating modes

Record mode and field mode are supported.

6.2.2.1 Keyboard

In addition to the data keys (alphabetic letters, numeric characters and special characters), BERMUDA supports three other categories of function keys:

- system function keys
- user function keys
- short message keys.

BERMUDA or the data display terminal perform validity checks on the key functions or the key combinations.

If the function is invalid, either BERMUDA or the display terminal will reject it and the appropriate message is displayed (see following tables).

As a result of rejection by BERMUDA, message display occurs in the BERMUDA indicator line or, in the case of terminal rejection, in the device status line.

The validity check ensues automatically and must not be programmed.

In the case of individual keys or key combinations, such as

- key combinations which activate user control characters,
- VALID key
- short message keys,

the activation or further treatment can be influenced by the programmer.

The measures to be taken in such cases are to be found in the appropriate tables containing the information.

<u>Note</u>

The abbreviations in the column 'Message to BERMUDA', of the table, have the following meaning:

(A): Application program is activated by BERMUDA.

(B): User procedure is activated by BERMUDA.

Further treatment of the message is influenced by the programmer.

(I): BERMUDA internal functions.

Further treatment of the message cannot be influenced by the programmer.

The abbreviations in the column 'Operating mode', of the table, have the following meaning:

SB: Record mode

FB: Field mode

System function keys

The system function keys are divided into the following groups:

- Correction functions
- Positioning functions
- Termination functions (in field mode only)
- User function keys

When these keys are pressed, fixed specified functions are activated that are executed immediately by the data display terminal or BERMUDA.

Those functions that are not supported in record mode can be seen in the following tables.

Correction functions

Key	Operating mode	Function	BERMUDA-valid	Message to BERMUDA	Local function in display terminal
<—	SB	Move one position to the left in field	X	-	X
	FB	As in SB	X	-	X
—>	SB	Move one position to the right in field	X	-	X
	FB	As in SB	X	-	X
LZF / LOEF	LZF / LOEF SB	Erase field	X	-	X
	LZF / LOEF FB	As in SB	X	-	X

Key	Operating mode	Function	BERMUDA-valid	Message to BERMUDA	Local function in display terminal
FRS	SB	Ineffec-tive			
	FB	Reset flasher	X	-	X
RS	SB	Reset terminal	X	-	X
	FB	Reset terminal	X	-	X
MAR KOR	MAR KOR SB	Mark field	-	-	X Display of 'Fehl' in device status line.
	MAR KOR FB	Character correc-tion in checking mode 2)	X	-	X
EFG	FB	Insert character	X	-	X
	SB	As in FB	X	-	X
AFG	FB	Delete character	X	-	X
	SB	As in FB	X	-	X
EM	FB	Set end marker	-	-	X Display of 'Fehl' in device stat-line.
	SB	Set end marker	- 2)	X	-
NUL	FB	Erase character	X	-	X
	SB	As in FB	X	-	X

Key	Operating mode	Function	BERMUDA-valid	Message to BERMUDA	Local function in display terminal
PAR	FB	Modify parameter range	X	-	X
	SB	As in FB	X	-	X
LZE	SB	Logical end of line	X	-	X
	FB	Field correction in checking mode 1)	X	X (I)	-
AFZ	FB	Delete line	-	-	- Display of 'Fehl' in device status line
	SB	As in FB	-	-	-
EFZ	FB	Insert line	-	-	- Display of 'Fehl' in device display line
	SB	As in FB	-	-	-
LSP	FB	Clear memory	-	-	- Display of 'Fehl' in device display line
	SB	As in FB	-	-	-

Key	Operating mode	Function	BERMUDA-valid	Message to BERMUDA	Local function in display terminal
↓	SB	Cursor down	X	-	X
	FB	-	- 1)	X 1)	-
<—<	SB	Position at first field of current line	X	-	X
	FB	-	- 1)	X 1)	-
<—┘	SB	Ineffective			
	FB	-	- 1)	X 1)	-

Table 6-13 Positioning functions

1) Activation of this function key does not produce a meaningful processing instruction for BERMUDA.

Activation of this key is rejected with the message
'FKT-TASTE UNZULAESSIG'
(invalid function key) in the BERMUDA indicator line.

If the current field, in field mode, is an empty mandatory field or a partly filled integrity field, activation of the key in such circumstances violates syntax rules (V or Z) [13] and is thus rejected, by means of a message in the BERMUDA indicator line.

Activation of an inadmissible function key in record mode is, in this case, rejected with the message
'FKT-TASTE UNZULAESSIG'
(invalid function key) in the BERMUDA indicator line.

After actuation of an invalid key, the cursor moves to the start of the current input field.

Termination Functions

Key	Operating mode	Function	BERMUDA-valid	Message to BERMUDA	Local function in display terminal
FE+	FB	End-of-field positive positive	X	X (I)	-
	SB	Ineffec-tive			
FE-	FB	End-of-field negative	X	X (I)	-
	SB	Ineffec-tive			
DÜ1 / SEND	DÜ1 / SEND SB	Send screen contents	X	X (A)	-
	DÜ1 / SEND FB	End-of-record	X	X (A)	-
F1/K1 / AKT	F1/K1 / AKT SB	See tables: System function keys and User function keys			
	F1/K1 / AKT FB	End of action	X	X (A)	-

Table 6-14 Termination functions

Operating Aids

Key	Operating mode	Function	BERMUDA-valid	Message to BERMUDA	Local function in display terminal
AM / DUPS	**AM** / DUPS SB	AM: Set start marker	X Rejected as syntax error in BERMUDA indicator line	X (I)	-
	AM / **DUPS** FB	DUPS: Set duplication switch	X	X (I)	-
DUP	SB	Ineffective			
	FB	Manual duplication	X	X (I)	-
VALID	SB	Ineffective	-	-	-
	FB	Syntax checking off	X 1)	-	X
RU / ANZ	**RU** / ANZ SB	-	-	-	X Display of 'Fehl' in device status line
	RU / **ANZ** FB	Display (in checking mode only 2)	X	X (I)	-
ESC / ;	SB	Set duplication switch (DUPS)	X	X (I)	-
	FB	As in SB 3)	X	X (I)	-

Key	Operating mode	Function	BERMUDA-valid	Message to BERMUDA	Local function in display terminal
VA	SB	Dis-connect; Ineffec-tive with dedicated line	X	-	X
	FB	As in SB	X	-	X

Table 6-15 Operating aids

1) Activation of this key does not effect message dispatch to BERMUDA. Syntax checking is deactivated locally in the data display terminal. The deactivation is communicated to BERMUDA with the next message and this is followed by the initiation of the VALID procedure by BERMUDA.

2) If the key is depressed at any other time than in checking mode, actuation is rejected with the message
'FKT-TASTE UNZULAESSIG'
(invalid function key) in the BERMUDA indicator line.

If, during field mode, the current field is an empty mandatory field or a partly filled integrity field, key actuation is rejected, owing to syntax error (V or Z), with message in the device status line.

Activation of an invalid function key in record mode is, in this case, rejected with the message
'FKT-TASTE UNZULAESSIG'
(invalid function key).

If an invalid key is pressed, the cursor is positioned at the start of the current input field.

3) The key sequence ESC; is in field mode equivalent to the DUPS key in field mode.

Key	Operating mode	Function	BERMUDA-valid	Message to BERMUDA	Local function in display terminal
EINF	SB	Ineffective			
	FB	Project-specific short message; Function with TRINIDAD: EINF	X	X (A)(B)	-
SATZ	SB	Ineffective			
	FB	Project-specific short message; Function with TRINIDAD: SATZ <	X	X (A)(B)	-
FORM	SB	Ineffective			
	FB	Project-specific short message; Function with TRINIDAD: FORM	X	X (A)(B)	-
LOES	SB	Ineffective			
	FB	Project-specific short message; Function with TRINIDAD: LOES	X	X	-

Key	Operating mode	Function	BERMUDA-valid	Message to BERMUDA	Local function in display terminal
ZW	SB	Ineffec-tive			
	FB	Project-specific short message; Function with TRINIDAD: ZW	X	X	-

Table 6-18 Short message keys (field mode)

<u>Note</u>

The specification of the procedure declaration YBKT can suppress the initiation of the short message function.

Further information concerning procedure declaration is to be found in the manual [13].

System and user function keys (record mode)

These function keys permit project-specific short messages (telegrams) to be sent to the application program.
The application program responds to key activation according to the declaration.
If the application program is the data entry service TRINIDAD, then the following logical short programs already have a fixed meaning [17].
When the keys are used for project-specific short messages, only the code of the actuated key and not data is received by the application program.L#
Further information regarding project-specific short messages [15].

Key	Operating mode	Function	BERMUDA-valid	Message to BERMUDA	Local function in display terminal
F3/K3 / DIAG	F3/K3 / DIAG SB	<u>Without SHIFT key</u> Is considered as user function key for initiating the project-specific short message K3 Function with TRINIDAD: SATZ <--	X	X (A)(B)	-
		<u>With SHIFT key</u> F3 is considered as system function key Initiation of F3 key is interpreted by BERMUDA like the diagnosis key	X	X (I)	-

Key	Operating mode	Function	BERMUDA-valid	Message to BERMUDA	Local function in display terminal
	F3/K3 DIAG FB	See table 6-16: Test and diagnosis			
F2/K2 TEST	F2/K2 TEST SB	Without SHIFT key Is considered as user function key for initiating the project-specific short message K2 Function with TRINIDAD: SATZ ——>	X	X (A)(B)	-
		With SHIFT key F2 is considered as the system function key. Initiation of F2 is interpreted by BERMUDA like the VALID key.	X	X (B)	-
	F2/K2 TEST FB	See table 6-16: Test and diagnosis			

Key	Operating mode	Function	BERMUDA-valid	Message to BERMUDA	Local function in display terminal
F1/K1 AKT	F1/K1 AKT SB	Without SHIFT key Is considered as user function key for initiating the project-specific short message K1 Function with TRINIDAD: ZW	X	X (A)(B)	-
		With SHIFT key F1 is considered as a system function key. Initiation of F1 is interpreted by BERMUDA like the AKT action control	X	X (A)	-
	F1/K1 AKT FB	See table 6-14: Termination functions			

Key	Operating mode	Function	BERMUDA-valid	Message to BERMUDA	Local function in display terminal
ESC V	SB	Project-specific short message K4; Function with TRINIDAD: LOES	X	X (A)(B)	-
	FB	As in SB	X 1)	X (A)(B)	-
ESC W	SB	Project-specific short message K5; Function with TRINIDAD: EINF	X	X (A)(B)	-
	FB	As in SB	X 1)	X (A)(B)	-
ESC M	SB	Project-specific short message K6; Function with TRINIDAD: FORM	X	X	-
	FB	As in SB	X 1)	X (A)(B)	-
ESC N	SB	Project-specific short message K7; Function with TRINIDAD: UNT	X	X (A)(B)	-
	FB	As in SB	X 1)	X (A)(B)	-

Key	Operating mode	Function	BERMUDA-valid	Message to BERMUDA	Local function in display terminal
ESC 0	SB	Project-specific short message K8; Function with TRINIDAD: SUCH	X	X (A)(B)	-
	FB	As is SB	X 1)	X (A)(B)	-

Table 6-19 System and user function keys (record mode)

1) These key combinations are generated, in record mode, by the 'UNT', 'ZW' keys etc.
'Manual input' is also a possible alternative to this.

Note

Project-specific short message initiation can be suppressed when the YBKT procedure declaration is specified.

Further information regarding Procedure declarations is to be found in the manual [13].

Invalid Key Combinations in Record Mode and Field Mode

Key	Operating mode	Function	BERMUDA-valid	Message to BERMUDA	Local function in display terminal
ESC ^	SB	Function code F4	- 1)	X (I)	-
	FB	As in SB	- 1)	X (I)	-
ESC ⌴	SB	Function code F5	- As with F4	- (I)	-
	FB	As in SB	- 1)	X (I)	-

Table 6-20 Invalid key combinations in record mode and field mode

1) No meaningful processing instruction is communicated to BERMUDA if these keys are pressed.

Key actuation is rejected with the message
'FKT-TASTE UNZULAESSIG'
(invalid key function) appearing in the BERMUDA indicator line.

If, in field mode, the current field is an empty mandatory field or a partially-filled integrity field, key actuation is rejected, owing to a syntax error
(V or Z) [12], with message appearing in the device status line.

Actuation of an invalid function key in record mode is, in this case, rejected with the message
'FKT-TASTE UNZULAESSIG'
(invalid key function) appearing in the BERMUDA indicator line.

If an invalid key is pressed, the cursor is positioned at the start of the current input field.

Further information concerning the keyboard can be found in the appropriate manuals [12], [13].

6.2.2.2 Indicator Line

BERMUDA supports output in the indicator line.

Status information and error text that the programmer has defined in the application program or in user procedures is displayed in this line.

Definition of status information in the application programs

APS application program:

- $YBSI procedure

COBOL application program:

- YBSI@@01

Definition of error text in the application programs

APS application program:

- $YBTX procedure

COBOL application program:

- YBTX@@01

Definition of status information and error text in user programs

Status information:

- $YBMS procedure

Error text:

- $YBER procedure

Further information concerning procedures and possibilities for their use [13].

6.2.2.3 Text Presentation, Field Attributes and Cursor Positioning

When formats are generated for the 9750 Data Display Terminal, text presentation and attributes for fields can be specified using the screen language MASK and the Interactive Format Generator IFG.

Format Generation with IFG

The formats are generated interactively and under the control of the user at the terminal. IFG is an application program and uses the Format Handling System FHS. IFG formats can be utilized in the terminal computer in APS and COBOL programs which use BERMUDA format handling.
IFG supplies a default user profile which can be modified according to the text presentation or field attribute that one requires.

Text Presentation

- Bright
- Normal
- Flashing
- Dark

Field Attributes

- Protected field; the field cannot be overwritten via keyboard.
- Printable field
- Numeric field (numeric input only)

Cursor Positioning

The cursor is positioned to a defined field.

Format Generation with MASK

Text Presentation with the Text Statement (T)

When an invariable text field is defined by the text statement, differing intensity levels can be assigned to the text:

- <u>H</u> - low intensity (default value)
- I - high intensity

The start of text and consequently that of the text presentation, in low intensity or high intensity, is specified in the text statement by means of the column and line numbers.

Example

1)	T01001␣'TRANSDATA␣960'	The text TRANSDATA 960 is displayed at low intensity in line 1 of the mask, beginning in column 1.
2)	T03010␣'BERMUDA'	The text BERMUDA is displayed at low intensity in line 3, beginning in column 10.
3)	T10005␣<I>␣'Name:'	The text Name: is displayed at high intensity in line 10, beginning at column 5.

Further information regarding the text statement [13].

Text Presentation with the Variables Statement (V)

Specification of attributes (optional) enables the presentation of the variable field (net data field) and the inputted characters to be controlled.

Attribute	Meaning	Presentation on the screen
S	Static field	Field does not flash (default value)
B	Flashing field	Characters that have been input are displayed flashing on the screen.
I	High-intensity field	Field is displayed at maximum brightness.
H	Low-intensity field	Field is displayed at reduced brightness.
D	Dark field	Characters entered do not appear on the screen.

Table 6-21 Attributes which control the screen display

The beginning of the variable field is determined by line and column values in the variables statement.

The effectiveness of the attributes depends on the length of the variable field.

The length is specified in the 'field definition' operand of the variables statement.

Example

1) V02008␣6 — A net data field 6 characters in length is defined, beginning in line 2, column 8. Default values are used for the attributes:

   - static field
   - high-intensity field

2) V10009␣4␣<SI> — A net data field 4 characters in length is defined, beginning in line 10, column 9 and having the following attributes:

   - static field
   - high-intensity field

3) V10015␣3␣<D> — A net data field 3 characters in length is defined, beginning in line 10, column 15 and having the following attribute:

   - dark field

4) V120007␣2␣<I,B> — A net data field of one character in length is defined, beginning in line 12, column 7 and having the following attributes:

   - high-intensity field
   - flashing field

Field Attribute Definition with the Variables Statement (V)

The field attributes, protected and unprotected, can be specified by means of attributes. These attributes control the keyboard inputs.

Attribute	Meaning	Presentation on the screen
U	Unprotected field	The user can input data to the field (default).
G	Protected field	The user cannot input characters to this field. The field can be provided with data by MASK or by user routines. The characters are displayed on the screen.

Table 6-22 Attributes which control keyboard input

Example

1) V10009˽4˽<S I U> — A net data field 4 characters in length is defined, beginning in line 10, column 9 and having the following attributes:

   - static field
   - high-intensity field
   - unprotected field

2) V200012˽4<I B G> — A net data field 4 characters in length is defined, beginning in line 20, column 12 and having the following attributes:

   - high-intensity field
   - flashing field
   - protected field

Further information concerning

- formats,
- the utilization of IFG and
- application preparation of BERMUDA can be found in manuals [13] and [16].

U1593-J-Z76-1-7600

7 SOFTWARE COMPONENTS IN HOST COMPUTERS

For supporting communication between the application program and the data display terminal, various BS2000 software components are available to the programmer.

These enable the user to program inputs and outputs from or at the data display terminals at virtual level. This applies to both the non-formatted and formatted modes at the terminals in the

- time-sharing mode,
- real-time processing mode and
- transaction mode.

Which of the software components can be used for each individual application is dependent on the operating mode and access method made use of by the respective application program.

The BS2000 components TIAM, DCAM as well as UTM have the following tasks:

- Transport and temporary storage of messages,
- Realization of the time-sharing mode, inquiry and transaction processing and transaction mode,
- Setting up and dismantling, as well as the management of, virtual connections,
- Display support at data display terminals,
- Management of the communication system.

These software components predefined interfaces to the application programs.

The software components

- IFG (Interactive Format Generator) and
- FHS (Format Handling System)

serve, at virtual level, to edit formatted data (messages) exchanged interactively between the application program and the data display terminal and support their input and output.

7.1 DCAM

7.1.1 Brief Description

DCAM (Data Communication Access Method) assists the communication of programs with each other within the same host computer and with data terminals.

The host computer works with the BS2000 Operating System.

DCAM supports the principle of distributed processing performance by means of communication from host computer to host computer i.e. mutual communication.

The tasks (programs) of one host computer can communicate with the tasks (programs) of another host computer.

DCAM supports communication with data display terminals via virtual terminals.

Further information concerning DCAM [19].

7.1.2 Technical Knowledge Required

Specialist knowledge is required in the following areas in order to program inputs and outputs on data display terminals:

- BS2000 Operating System
- COBOL and ASSEMBLER programming languages
- DCAM, COBOL calls [20]
- DCAM, macro calls [21]
- DCAM, program interfaces [19]
- Physical programming of data display terminals (see chapter 4).

7.1.3 Supported Data Display Terminals

All data display terminals are supported in accordance with Table 2-1.

Notes

It should be noted, with regard to generation in PDN, that

- the 9749 Data Display Terminal must be generated as the 9750 Data Display Terminal and
- the 9750-62 Data Display Terminal must be generated as the 8162 Data Display Terminal.

7.1.4 Access to Data Display Terminals

The programmer can choose between two possibilities:

- physical programming
- utilization of virtual terminals.

In the case of physical programming, the programmer has to construct the message (message headers, parameter ranges and device-specific device functions) at the data display terminal himself.

Utilization of virtual terminals relieves the programmer of the physical programming.

7.1.5 Virtual Terminals

The programmer can choose between two virtual terminals:

- line terminal
- format (form) terminal

When a connection is established (YOPNCON), it is possible for the programmer to determine if the line terminal or the form terminal is to be used.

Further information concerning

- line terminals,
- format terminals and
- establishment of connections

see [19], [20], [21].

7.1.6 Support for ASSEMBLER and COBOL Programs

Input and output in ASSEMBLER and COBOL programs is supported.

Consequently, only those operands, structures and logical control characters are addressed, whose specification is required so that the system

- initiates a terminal function (output)

or

- responds to terminal function (input).

All preparatory measures are assumed to be known.

7.1.6.1 ASSEMBLER Programs

<u>Input and output</u>

EDIT=SYSTEM	This specification is required if virtual terminals are to be used.

<u>Input</u>

EDITIN=LINE (standard)	The system edits messages with the help of virtual line terminals.
EDITIN=GETFC	This operand must then be specified, if the logical function key code is to be transferred as the first character of the message.
EDITIN=NGETFC	The logical function key code is not transferred.

<u>Output</u>

EDITOUT=LINE (standard)	The system edits messages with the help of the virtual terminals.

<u>Audible alarm:</u>

If an audible alarm is to be triggered at outputting (MACRO YSEND), OPTCD=BELL must be specified in the RPB control block (Request Parameter Block).

<u>Notes</u>

The macro call YCHANGE can be used to modify characteristics of a connection established by YOPNCON [21].

Further information concerning the macros YCCB and YOPNCON [21].

VTCSET macro call

Logical control characters can also be specified in the form of symbolic names which are generated by the macro call VTCSET.
Using these names, logical control characters can be inserted in line mode output messages and be located in line mode input messages.

Physical programming of the message

If the programmer has to edit messages himself e.g. for terminal functions that are not supported by the system, the following specifications are to be made:

- EDIT=SYSTEM
- EDITIN=PHYS (during input)
- EDITOUT=PHYS (during output)

Further information [21].

7.1.6.2 COBOL Programs

The following structures must be supplied:

V structure

Input and output

EDIT "SYS"	This specification is required if virtual terminals are to be used.

Input

EDITIN "LIN"	The system edits messages with the help of the virtual line terminals.
GETFC "YES"	This operand must then be specified, if the logical function key code is to be transferred as the first character of the message.
GETFC "NO"	The logical function key code is not transferred.

<u>Output</u>

EDITOUT "LIN"	The system edits messages with the help of the virtual terminals.

B structure

<u>Input and output</u>

FHS "YES"	The formatted input and output of messages takes place using the format control FHS (Format Handling System).

If the user wants to input or output messages by means of FHS=YES, he must supply the following fields, either when connection is established with macro YOPNCON or using the function 'Modifying the characteristics of a connection (YCHANGE)':

- EDIT with "SYS" for message editing by the system.
- EDITIN with "FOR" for format control during input.
- EDITOUT with "FOR" for format control during output.

Further information concerning FHS "YES" [21].

<u>Notes</u>

The YCHANGE call can modify characteristics of a connection established with YOPNCON [21].

Copy member TIAMTRC

The copy member TIAMTRC contains logical control characters for the line-mode (EDIT "SYS" and EDITOUT "LIN" or EDITIN "LIN").
Copying to the user program can be done using COPY TIAMTRC.

Physical programming of the message

If the programmer has to edit messages himself, e.g. terminal functions not supported by the system, the following specifications are to be made:

- EDIT "SYS"
- EDITIN "PHY" (for input),
- EDITOUT "PHY" (for output).

Further information [21].

Physical programming (see chapter 4).

7.1.6.3 Logical Control Characters

Input

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Logical end of line	prefix NL	NEW-LINE	X'15'

Effect

The logical end of line is represented by the device-independent control character X'15' with regard to messages to the program.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Only at the 9750-62: switch over to 2nd character set	prefix SO	SECOND-CHAR-SET	X'0E'

Effect

The subsequent text characters are characters of 2nd character set.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Switch over to normal character set	prefix SI	NORMAL-CHAR-SET	X'0F'

Effect

The subsequent text characters are characters of the normal character set.

7

Output

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Logical end of line	prefix NL	NEW-LINE	X'15'

Effect

- The cursor is set to the beginning of the next line.
- Text handling is reset to
  - not flashing,
  - not underscored/not reverse,*
  - normal intensity.

    9752 Data Display Terminal:

    Normal intensity is displayed green.
- Logical end of line is outputted.
- Field handling is reset to unprotected
- 9750-62 Data Display Terminal:

  If the 2nd character set was on, a switchover to normal character set is effected.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
New page	prefix NP	NEW-PAGE	X'0C'

Effect

- The screen contents are erased.
  The cursor is set to start of screen.
- Text handling is reset to
  - not flashing,
  - underscored/reverse,*
  - with normal intensity,

    9752 Data Display Terminal:

    Normal intensity is displayed green.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Emphasized layout 1	prefix EM1	EMPH-LAYOUT1	X'1D'

Effect

The subsequent text characters are displayed

- with reduced intensity,
- flashing,
- not underscored/not reverse,*

  9752 Data Display Terminal:
  Reduced intensity is displayed yellow.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Emphasized layout 2	prefix EM2	EMPH-LAYOUT2	X'1F'

Effect

The subsequent text characters are displayed

- with reduced intensity,
- underscored/reverse,*
- not flashing.

  9752 Data Display Terminal:
  Reduced intensity underscore are displayed white.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Emphasized layout 3	prefix EM3	EMPH-LAYOUT3	X'13'

Effect

The subsequent text characters are displayed

- with normal intensity,
- not underscored/not reverse,*
- not flashing.

  9752 Data Display Terminal:
  Normal intensity is displayed green.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Emphasized layout 4	prefix EM4	EMPH-LAYOUT4	X'14'

Effect

The subsequent text characters are displayed

- with normal intensity,
- underscored/reverse,*
- not flashing.

  9752 Data Display Terminal:

  Normal intensity and underscore are displayed red.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Normal layout	prefix NOR	NORMAL-LAYOUT	X'1E'

Effect

The subsequent text characters are displayed

- with reduced intensity,
- not underscored/not reverse,*
- not flashing.

  9752 Data Display Terminal:

  Reduced intensity is displayed yellow.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Switchover to the 2nd character set	prefix SO	SECOND-CHAR-SET	X'0E'

Effect

9750-62 Data Display Terminal:

The subsequent text characters are outputted in character set 2.

Function	Symbolic name ASSEMBLER	Symbolic name COBOL	Coding (hexadec.)
Switchover to normal character set	prefix SI	NORMAL-CHAR-SET	X'0F'

Effect

9750-62 Data Display Terminal:

The subsequent text characters are displayed in character set 1.

Function	Symbolic name ASSEMBLER	Symbolic name COBOL	Coding (hexadec.)
Start protected area	prefix SPA	START-PROT-AREA	X'36'

Effect

- The following field is defined as protected.
- Subsequent text characters are displayed with reduced intensity.

  9752 Data Display Terminal:

  Reduced intensity is displayed yellow.

Function	Symbolic name ASSEMBLER	Symbolic name COBOL	Coding (hexadec.)
End area protection	prefix EPA	END-PROT-AREA	X'08'

Effect

- The field following is unprotected.
- Subsequent text characters are displayed with reduced intensity.
- 9752 Data Display Terminal:

  Reduced intensity is displayed yellow.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Unprotect numerical field	prefix NUM	——	X'11'

Effect

Only numerical inputs can be made in the subsequent field.
The text characters are displayed with reduced intensity.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec)
Display is suppressed	prefix DAR	——	X'12'

Effect

The subsequent screen section is controlled blank.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec)
Delete character	prefix DEL	——	X'07'

Effect

The character is removed from the output text and is not passed on to the display terminal.

* The screen presentation in such cases depends on the setting of the hardware [7] in the device.

7.2 TIAM

7.2.1 Brief Description

TIAM (Terminal Interactive Access Method) is a BS2000 component for dialog in the time-sharing mode. It supports in conjunction with BCAM (Basic Communication Access Method) communication between application programs in host computers and the terminals connected to them.
The dialog between the terminal user and the BS2000 computer of a network proceeds interactively through TIAM and BS2000 commands.
As the user operates in the system mode all the BS2000 services are available.
Further information about TIAM commands (see [23]).
Besides working in the system mode, the user can also use TIAM in an alternative way with the so-called program mode. Using the latter method, the user works with a program that he has called in the system mode via an EXEC or LOAD command. This can be, for example, a utility program or a self-made application program.

To generate such programs, TIAM offers the programmer

- macro calls (ASSEMBLER) and
- COBOL calls.

By employing the TIAM macro calls, the programmer can

- define and modify data display terminal characteristics and
- test task values.

Using TIAM-COBOL calls, the programmer can control input/output.

In order to program the 9749, 9750 and 9752 Data Display Terminals, knowledge of the following is required:

- BS2000 Operating System,
- Assembler or COBOL programming language,
- Physical programming of data display terminals.

7.2.2 Supported Data Display Terminal Versions

All versions of the 9749, 9750 and 9752 Data Display Terminals are supported in accordance with table 2-1.
All versions of the data display terminals contained in table 2-1 are supported.

7.2.3 Access to the Data Display Terminals

The programmer can choose between two possibilities:

- physical programming,
- programming employing the "logical terminal".

With regard to physical programming, the programmer has to construct the message (message header, parameter ranges and specific device functions optional) for the data display terminal himself.
The employment of "logical terminals" enables the programmer to be freed from the physical programming thus making his work considerably easier.

7.2.4 Virtual Terminals

The programmer can choose between two logical terminals:

- the line terminal and
- the format terminal.

7.2.5 Support for Data Display Terminals

The support is carried out for input and output in ASSEMBLER and COBOL programs.
Consequently, only the operands, structures and logical control characters are addressed whose specification is required in order that the system can effect the following:

- initiate a terminal function (output) or
- respond to a terminal function (input).

All preparatory measures are assumed to be known.

7.2.5.1 Support for Programming in ASSEMBLER

The support by means of TIAM for programming in ASSEMBLER is divided in support for

- Input (RDATA macro call),
- Combined input/output (WRTRD macro call),
- Output (WROUT macro call),

as well as supporting

- the changing of the characteristics of data display terminals (TCHNG macro call),
- the requesting of display terminal characteristics (TSTAT macro call),
- Inputs and outputs by means of logical control characters (VTCSET macro call).

RDATA macro call

The user program can, using the macro RDATA, read a variable-length record (message) from the terminal carrying out the task.

<u>WRTRD macro call</u>

The user program can, using the macro WRTRD, send a message to the display terminal in interactive mode and immediately afterwards read a message from this terminal.

<u>WROUT macro call</u>

The macro WROUT transfers a message in interactive mode from the user program to the display terminal carrying out the task.

<u>TCHNG macro call</u>

The user program can, using the TCHNG macro, change the characteristics of the virtual terminal.

TCHNG remains effective until the user program in question is terminated and applies only for the inputs and outputs of this program, via the macros RDATA, WROUT and WRTRD, at the user's terminal.

<u>TSTAT macro</u>

With the aid of the TSTAT macro, the user can request information on the display terminal in timesharing mode, such as

- TYPE and characteristics of the display terminal,

- Terminal and processor names etc.

Further information regarding macro calls [23].

<u>VTCSET macro call</u>

The macro VTCSET generates symbolic names with which logical control characters (record control characters and display control characters) can be inserted in line mode output messages or be identified in line mode input messages.

The logical control characters generated by VTCSET and used in ASSEMBLER programming are contained in the section "Logical Control Characters for Input and Output".

Further information regarding the VTCSET macro call [24].

Physical Programming of Message Editing

If the programmer has to carry out message editing himself, e.g. in the case of terminal functions that are not supported by the system, then MODE=PHYS must be specified in the macro calls

- WROUT (Output) and
- WRTRD (Combined input/output).

Further information regarding physical programming (see chapter 4).

7

7.2.5.2 Support for Programming in COBOL

The TIAM COBOL interface offers the following functions for supporting input and output to the COBOL programmer:

- RDATA (Data input)
- WROUT (Data output)
- WRTRD (Combined data input/output)

These functions are implemented as subroutines (CALL...).
Format handling with the aid of FHS is integrated as a special function into the calls CALL "WROUT" and CALL "WRTRD".

COBOL call CALL "RDATA"

The user program can read a record from the terminal carrying out the task, using the call CALL "RDATA".

COBOL call CALL "WROUT"

The call CALL "WROUT" enables the COBOL programmer to output a message to the display terminal in timesharing mode.

COBOL call CALL "WRTRD"

The call CALL "WRTRD" enables the COBOL programmer, in timesharing mode, to output a message to the terminal and immediately afterwards read a message from this terminal.
Further information concerning the COBOL calls CALL "RDATA", CALL "WROUT", CALL "WRTRD" [23].

7.2.5.3 Data Structures for the TIAM COBOL Calls

Before issuing any input/output calls, the programmer must supply certain data structures with information for TIAM regarding the desired I/O. These data structures enable communication between TIAM and the user program. They are contained in a library as COPY elements and are copied into the user program.

The following data structures are available:

- TIAM-CONTROL-INFO
  controls the input/output,
- LINE-MODE-CONTROL-CHARACTERS
  contains logical record control characters required in line mode output (logical line terminal).

Data structure TIAM-CONTROL-INFO

The data structure TIAM-CONTROL-INFO is copied into the user program by means of the call
COPY TIAMINFO.

Data structure LINE-MODE-CONTROL-CHARACTERS

Data structure LINE-MODE-CONTROL-CHARACTERS is copied to the user program by means of call
COPY TIAMCTRC.

The programmer can transfer the logical control characters which he requires for the input/output support and which are contained in this data structure by means of a simple MOVE instruction into the appropriate ranges.
These logical control characters used in COBOL programming are covered in the section "Logical Control Characters for Input and Output".

Further information regarding data structures for TIAM COBOL calls [23].

Physical Programming of Message Editing

If the programmer has to edit messages himself, e.g. in order to use display terminal functions that are not supported by the system, then the following specifications are required in the data structure TIAM-CONTROL-INFO:

- EDIT-OUT "P"
- EDIT-IN "P"
- EDIT-MODE "P"

Further information regarding physical programming (see chapter 4).

7.2.5.4 Logical Control Characters for Input and Output

Listed below are the functions of the logical control characters (record and display control characters) for the input and output.

In the case of **ASSEMBLER programs,** the control characters must be generated by means of the VTCSET macro.

For **COBOL programs,** the control characters are contained in the data structure
LINE-MODE-CONTROL-CHARACTERS.

The data structure must be copied to the user program by means of the copy element TIAMTRC.

Input

Function	Symbolic name ASSEMBLER	COBOL	Coding (hex)
Logical end of line	prefix NL	NEW-LINE	X'15'

Effect

The logical end of line is represented by the device-independent control character X'15' with regard to messages to the program.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hex)
Only at the 9750-62: switchover to 2nd character set	prefix SO	SECOND-CHAR-SET	X'0E'

Effect

The subsequent text characters are characters of 2nd character set.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hex)
Switchover to normal character set	prefix SI	NORMAL-CHAR-SET	X'0F'

Effect

The subsequent text characters are characters of the normal character set.

Output

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Logical end of line	prefix NL	NEW-LINE	X'15'

Effect

- The cursor is set to the beginning of the next line.
- Text handling is reset to
  - not flashing,
  - not underscored/not reverse,*
  - normal intensity.

    9752 Data Display Terminal:

    Normal intensity is displayed green.
- Logical end of line is outputted.
- Field handling is reset to unprotected.
- 9750-62 Data Display Terminal:

  If the 2nd character set was on, a switchover to normal character set is effected.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
New page	prefix NP	NEW-PAGE	X'0C'

Effect

- The screen contents are erased.
  The cursor is set to start of screen.
- Text handling is reset to
  - not flashing,
  - underscored/reverse,*
  - with normal intensity,

    9752 Data Display Terminal:

    Normal intensity is displayed green.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Emphasized layout 1	prefix EM1	EMPH-LAYOUT1	X'1D'

Effect

The subsequent text characters are displayed

- with reduced intensity,
- flashing,
- not underscored/not reversed.*

9752 Data Display Terminal:

Reduced intensity is displayed yellow.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Emphasized layout 2	prefix EM2	EMPH-LAYOUT2	X'1F'

Effect

The subsequent text characters are displayed

- with reduced intensity,
- underscored/reverse,*
- not flashing.

9752 Data Display Terminal:

Reduced intensity and underscore are displayed white.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Emphasized layout 3	prefix EM3	EMPH-LAYOUT3	X'13'

Effect

The subsequent text characters are displayed

- with normal intensity,
- not underscored/not reverse,*
- not flashing.

9752 Data Display Terminal:

Normal intensity is displayed green.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Emphasized layout 4	prefix EM4	EMPH-LAYOUT4	X'14'

Effect

The subsequent text characters are displayed

- with normal intensity,
- underscored/reverse,*
- not flashing.

9752 Data Display Terminal:

Normal intensity and underscore are displayed red.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Normal layout	prefix NOR	NORMAL-LAYOUT	X'1E'

Effect

The subsequent text characters are displayed

- with reduced intensity,
- not underscored/not reverse,*
- not flashing.

9752 Data Display Terminal:

Reduced intensity is displayed yellow.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Switchover to the 2nd character set	prefix SO	SECOND-CHAR-SET	X'0E'

Effect

9750-62 Data Display Terminal:

The subsequent text characters are outputted in character set 2.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Switchover to normal character set	prefix SI	NORMAL-CHAR-SET	X'0F'

Effect

9750-62 Data Display Terminal:

The subsequent text characters are displayed in character set 1.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Start protected area	prefix SPA	START-PROT-AREA	X'36'

Effect

- The field following is defined as protected.
- Subsequent text characters are displayed with reduced intensity.

  9752 Data Display Terminal:

  Reduced intensity is displayed yellow.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
End area protection	prefix EPA	END-PROT-AREA	X'08'

Effect

- The field following is unprotected.
- Subsequent text characters are displayed with reduced intensity.
- 9752 Data Display Terminal:

  Reduced intensity is displayed yellow.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Unprotected numerical field	prefix NUM	——	X'11'

Effect

Only numerical input is possible in the subsequent field.

The text characters are displayed with normal intensity.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Display is suppressed	prefix DAR	——	X'12'

Effect

The subsequent screen section is controlled blank.

Function	Symbolic name ASSEMBLER	COBOL	Coding (hexadec.)
Delete character	prefix DEL	——	X'07'

Effect

The character is removed from the output text and is not passed on to the display terminal.

* The screen presentation in such cases depends on a hardware setting within the device [7].

7.2.5.5 Functions Supportet for ASSEMBLER and COBOL Programming

In tabular form below is a selection of terminal functions that TIAM supports for input/output purposes in conjunction with ASSEMBLER or COBOL programming.

The specifications to be made by the programmer in ASSEMBLER or COBOL in order to implement these functions are presented briefly in the appropriate columns.

For further information [18].

7

Input

Data display terminal function	Macro specification for input/output, with ASSEMBLER	Specification in the TIAM-CONTROL-INFO, with COBOL
The actual terminal should be treated as a logical line terminal. The message can be structured with the aid of logical control characters.	Macro RDATA: MODE=LINE	EDIT-IN EDIT-MODE "L"
Lower case characters are passed to the user program.	Macro RDATA: ILCASE=YES	EDIT-IN EDIT-OPTIONS "LCASE"
The logical code of the transmission key is to be sent as the first character of the message.	Macro RDATA: IGETFC=YES	EDIT-IN EDIT-OPTIONS "GETFC"
The input data is to be read from a defined badge reader connected to the display terminal.	Macro RDATA: IGETIC=YES	EDIT-IN EDIT-OPTIONS "GETIC"
The input data is confidential and must remain invisible at the terminal.	Macro RDATA: ICFD=YES	EDIT-IN EDIT-OPTIONS "CFDATA"

Output/input combined and output

Data display terminal function	Macro specification for input/output, with ASSEMBLER	Specification in the TIAM-CONTROL-INFO with COBOL
The user program is supported by format control (software component FHS) which edits the message in a form suitable for input/output to the terminal.	Macro WRTRD and WROUT: MODE=FORM	EDIT-OUT EDIT-MODE "F"
The message is to be physically (without being edited by the system) output or read in at the terminal. This allows special device functions to be executed for which the LINE or FORM mode is insufficient.	Macro WRTRD and WROUT: MODE=PHYS	EDIT-IN and EDIT-OUT "P"

Data display terminal function	Macro specification for input/output, with ASSEMBLER	Specification in the TIAM-CONTROL-INFO with COBOL
The output message to a display terminal is to be outputted simultaneously to a hard copy device (printer) connected to the terminal.	Macro WRTRD and WROUT: OHCOPY=YES	EDIT-OUT EDIT-OPTIONS "HCOPY"
The message is to be outputted unstructured (homogeneously) so that it is regarded as one output unit. Effect at the 9749, 9750 and 9752 Terminals in operating mode 1 is the following: The entire message can be transmitted back again by modifying one character in an output message.	Macro WRTRD and WROUT: OHOM=YES	EDIT-OUT EDIT-OPTIONS "HOMOUT"
The message is to be output as a structured message, i.e. in non-homogeneous form so that one logical line is regarded as the output unit. Effect at the 9749, 9750 and 9752 Terminals in operating mode 1 is as follows: Individual logical lines can be modified separately and thus transmitted (read) back selectively.	Macro WRTRD and WROUT: OHOM=NO	
Message output is announced by means of an audible signal (gong) at those display terminals fitted out with a special device extra suitable for the purpose.	Macro WRTRD and WROUT: OBELL=YES	EDIT-OUT EDIT-OPTIONS "BELL"
Lower case letters are passed to the user program as well.	Macro WRTRD: ILCASE=YES	EDIT-IN EDIT-OPTIONS "LCASE"
The logical code of the data transmission function keys is to be sent also, as the first character in the message.	Macro WRTRD: IGETFC=YES	EDIT-IN EDIT-OPTIONS "GETFC"
The input data is to be read from a defined badge reader connected to the terminal.	Macro WRTRD: IGETIC=YES	EDIT-IN EDIT-OPTIONS "GETIC"
The input data are confidential and must remain invisible at the terminal.	Macro WRTRD: ICFD=YES	EDIT-IN EDIT-OPTIONS "CFDATA"
The message is to be displayed in a special information line at the terminal, without destroying the screen contents.	Makro WROUT: OINFO=YES	EDIT-OUT EDIT-OPTIONS "INFO"

Table 7-1 Supported functions

7.3 FORMAT HANDLING SYSTEM

7.3.1 Brief Description

The Format Handling System helps the programmer to realize a forms-controlled dialog between the terminal and the application program.

FHS makes programs independent of the physical characteristics of terminals. The user can work with different terminals without having to be familiar with their varying physical characteristics.

In the case of output, FHS assembles a message by means of the previously defined format description and the data and attributes defined in the application program.
The message to the data display terminal comprises text and terminal-related control characters.

In the case of input, FHS removes the control characters contained in the message, together with the text fields not required by the application program.
FHS supplies the application program with variable fields only.

FHS can be utilized in all TIAM, DCAM and UTM programs that are written in Assembler, COBOL or RPG.
UTM and RPG use internal interfaces for this purpose.

It is recommended to use the Interactive Format Generator (IFG) to generate formats for the data display terminals.

Further information concerning IFG [22] and FHS [18].

7.3.2 Support for the 9749, 9750, 9752 Data Display Terminals

The supported functions and field attributes, as well as the specification involved, are listed in the following tables.

Defining the Control Block

When the control block is defined using the macro MDCBL, the following terminal functions can be specified:

Terminal function	Specification with ASSEMBLER in MDCBL or MUCBL	Specification with COBOL in FHS-MAIN-PAR
Audible alarm "BEL"	BEL=YES	MAP-BELL-OPTION "Y"
Read unprotected fields	IST=RUNP	MAP-READ-METHOD "RUNP"
Read modified fields	IST=RMOD	MAP-READ-METHOD "RMOD"
Read with NUL characters or with relevant NUL characters	NILS=YES	MAP-READ-NILS "Y"
Read without NUL characters	NILS=NO	MAP-READ-NILS "N"
Message is output automatically on hardcopy device and on screen.	AUTOHC=YES	MAP-AUTO-HARD-COPY "Y"
No automatic hardcopy mode	AUTOHC=NO	MAP-AUTO-HARD-COPY "N"
Clear screen before output of format	CLEAR=YES	MAP-CLEAR-OPTION "Y"
Overwrite screen; Screen contents not overwritten by new format remain	CLEAR=NO	MAP-CLEAR-OPTION "Y"
Keyboard is locked	KEYLOCK=YES	MAP-LOCK-KEYS "Y"
Keyboard is not locked	KEYLOCK=NO	MAP-LOCK-KEYS "N"
Automatic tabulator	ATAB=YES	MAP-AUTO-TAB "Y"
No automatic tabulator	ATAB=NO	MAP-AUTO-TAB "N"
Erase variable data on screen	MSTD=RESET	FHS-MAPPING-METHOD "RESET"
Local hardcopy mode	DEVICE=appropriate terminal HCOPY=LOCAL AUTOHC=YES	MAP-DEVICE-CLASS "DSS" MAP-HARDCOPY-OPTION "L" MAP-AUTO-HARDCOPY "Y"
At output, fields that are modified or marked are not reset	PMOD=YES	MAP-SCREEN-PRE-MOD "Y"

Table 7-2 Specification in the control block

Note

The control block can be modified with the MUCBL macro.

Modification of Field and Display Attributes of a Data Field

The field and display attributes of the format definition can be modified by means of:

- the MATUP macro or
- COBOL CALL FHSATTR or
- COPY element FHS-ATTRIBUTE-MOVE.

Prerequisites:

- Attribute fields must be available in the addressing aids.

In the case of ASSEMBLER programs

- MODY=YES (MUCBL macro)

In the case of COBOL programs

- FHS-MODY-ATTRS "Y" (FHS-MAIN-PAR)

Terminal function	Attribute with MATUP	Specification in FHS-ATTR-PAR
Unprotected field	UNPROT	A-PROT-LEVEL "UNPR"
Protected field	PROT	A-PROT-LEVEL "PROT"
Protected field; Is returned to host at input	PROTRET	A-PROT-LEVEL "PRET"
Normal (high-intensity 1)	BRT	A-DISP-LEVEL "B"
Reduced intensity 1)	NORM	A-DISP-LEVEL "H"
Dark field 1)	DRK	A-DISP-LEVEL "D"
Data field flashes 1)	SIGN [ON]	SIGNAL "Y"
Italics 1)	ITAL	A-ITALIC "Y"
Data field flashes after detection	DET	A-PROT-LEVEL "PDET"
Numeric field	NUM	A-NUMERIC "Y"
Data field is printable	PRINT	Printable is default value
Data field is not printable	NOPRINT	A-NO-HARDCOPY "Y"
Cursor positioning	IC; In MCMAP, value to operand CURSOR	<u>Specification in FHS-MAIN-PAR:</u> FHS-MAP-CURSOR-OPT "Y"; CALL "FHSCURS"

1) See display attributes of the various data display terminals

Table 7-3 Modification of field attributes and display attributes

7.3.2.1 Display Attributes of Data Display Terminal Versions

Display attribute specified in IFG	Attribute	Display on data display terminal 9750	9752
Bright	BRT	Bright	Green
Normal	NORM	Normal	Yellow
Dark	DRK	Not visible	Not visible
Flashing	SIGNON	Flashing	Flashing
Bright and italic script	BRT and ITAL	Bright and underscored	Red
Normal and italic script	NORM and ITAL	Normal and underscored	White

Table 7-4 Display attributes of data display terminal versions

7.4 INTERACTIVE FORMAT GENERATOR (IFG)

7.4.1 Brief Description

The Interactive Format Generator facilitates the generation of screen formats (masks).

The formats are generated interactively at a data display terminal with the the assistance of user prompting.

Formats generated by IFG can be used with

- the Format Handling System FHS in the host computer or
- BERMUDA format handling in the terminal computer [16].

IFG formats can be utilized with FHS in

- COBOL programs,
- RPG programs,
- ASSEMBLER programs.

These programs have to employ one of the following access methods:

- TIAM
- DCAM
- UTM
- ASMUS2
- BERMUDA.

7.4.2 Data Display Terminal Support

This chapter describes the supporting of field attributes (text presentation and field handling) and cursor positioning for the 9749, 9750 and 9752 Data Display Terminals in the form of an overview only.

Note

In IFG Version 3.0, the default values do not take special account of the 9749 and 9752 Data Display Terminals.

The format assignment for the 9750 Data Display Terminal is, however, compatible with the 9749 and 9752 Data Display Terminals.

Text Presentation

IFG enables text to be displayed in the following ways:

Specification in IFG	Display on the data display terminals 9749, 9750	9752
Bright	Bright	Green
Normal	Normal	Yellow
Dark	Dark	Dark
Italic script and bright	Bright and underscored or reverse *	Red
Flashing	Flashing	Flashing
Normal and italic script	Normal and underscored or reverse *	White

* Whether underscored or reverse depends on a hardware setting within the device [7]

Table 7-5 Text presentation in IFG

Field Attributes

The following field attributes can be defined with IFG:

- Protected field,
- Numeric field,
- Printable field,
- Detectable field,
- Automatic input.

Cursor Positioning

When "CURSOR: YES" is specified, the cursor is positioned to a defined field mode:

Notes

Note should be taken of the following attributes in the multiple field mode:

- The field attributes bright, normal, dark and italic script are ignored during format application.
- Bright is defined for "unprotected" fields.
- Normal is defined for "protected" fields.
- IFG checks that field delimiters are adhered to (max. 48).

Further information concerning the multiple field mode [22].

Correlation exists between the processing attributes of the IFG field, allocated the name 'Automatic input' (see 'Allocating field names/processing attributes) [22], and the corresponding FHS operands in the MDFLD macro [18].

With protected fields:

Automatic input 'YES' : ATTR=PROTRET

Automatic input 'NO' : ATTR=PROT

With unprotected fields:

Automatic input 'YES' : ATTR=FSET

Automatic input 'NO' : ATTR=UNPROT

Important information for the programmer:

The display attributes defined with IFG are equivalent to those specified in FHS with the ATTR operand in the MDFLD macro.

Correlation between the display attributes of IFG fields and the corresponding FHS operands:

Attribute	ATTR=
Bright	BRT
Normal	NORM
Dark	DRK
Protected	YES: PROT, NO: UNPROT
Numeric	NUM
Italic script	ITAL
Flashing	SIGN
Detectable	DET
Printable	PRINT
Cursor	IC

Further information about IFG [22] and FHS [18].

7.5 UTM

What does UTM do?

The Universal Transaction Monitor UTM controls transaction applications which process jobs in the interactive mode. Data display terminals and printers (if applicable) are linked to a transaction application (=UTM application).

UTM applications

- process (concurrently) a number of jobs in the interactive mode (transaction processing),
- have their own logical network of data terminals (independent administration),
- are simple to program (in ASSEMBLER, COBOL and RPG),
- can operate using formats via an integrated FHS interface,
- offer protection against misuse and the loss of data,
- allow automatic and programmed error handling,
- give the user access to files and data bases.

To facilitate the creation of these UTM application programs UTM offers the compatible data communication interface KDCS.

For further information about UTM, see [28], [29], [30], [32].

7.5.1 Data Terminal Support

This section is concerned only with the support of the 9749, 9750 and 9752 Data Display Terminals.

For information about which other data terminals are supported, the user should consult [30].

The following factors should be borne in mind at generation time:

All data terminals must be generated as the same station type in UTM and PDN/BCAM. This also applies to the 9749 Data Display Terminal, for which no special terminal type can be allocated in PDN and UTM.
Instead, the 9749 DDT is generated as if it were a 9750 device.

Further information on generation may be found in [30].

7.5.2 Support of Data Display Terminal Functions

The following display terminal functions are logically supported in UTM:

<u>In line mode via the macros VTCSET or TIAMCTRC</u>

Delete character
Blanking (video suppression)
Numeric input
Absolute positioning on a line
Positioning on the next line
Standard character set (9750-62 only)
2nd character set (9750-62 only)
Intensity
Colors: green, yellow, red and white (9752 only)
Clear screen
Protected field
Unprotected field
Flashing
Non-flashing
Underscore/reverse [7]

For further information about the macros VTCSET and TIAMCTRC, see [21], [23].

<u>In format mode via FHS</u>

Blanking (video suppression)
Numeric input
Intensity
Colors: green, yellow, red and white (9752 only)
Clear screen
Protected field
Unprotected field
Flashing
Non-flashing
Underscore/reverse [7]

For further information on FHS, see [18].

In both line and format modes it is additionally possible to specify screen output functions which support the following data display terminal functions:

DDT function	Mode
Trigger BEL function	L, F
Clear screen and output new format	F
Extended-line mode	L
Erase unprotected fields	F
Request insertion of badge (ID card)	L

L = Line mode
F = Format mode

Table 7-6: Functions of the DDT in line and format mode

Logical Support with Screen Output Functions

Screen output functions are presented directly to the UTM interface as UTM functions. The program implements them by means of an entry in the KDCS parameter field KCDF.

UTM supports the following functions:

DDT function	Entry in KDCS parameter field KCDF
Trigger BEL function	KCALARM
Clear screen and output new format	KCREPL
Extended-line mode	KCEXTEND
Erase unprotected fields	KCERAS
Request insertion of badge (ID card)	KCCARD

Table 7-7: Screen output functions

If there is no entry in KCDF (KCDF=LOW-VALUE), then UTM decides whether

- to clear the screen in the case of formatted output or
- allow only the modified data of a format to be overwritten without erasing the whole format.

Logical Support in Line Mode

In the line mode the programmer has certain logical control characters at his disposal.

In the case of ASSEMBLER programs, the control characters are copied to the application program with the macro VTCSET (see [23]).

In the case of COBOL programs, the control characters are copied to the application program with the COPY member TIAMCTRC (see [23]).

Certain control characters are only valid in conjunction with the screen output function KCEXTEND (see next table).

The abbreviations in the table have the following meanings:

NI = Normal intensity
RI = Reduced intensity
GR = Green
YE = Yellow
WH = White
RE = Red
* = Meaningful only in conjunction with the screen output function KCEXTEND
x = Function implemented via this control character
1) = 9750-62 only

Entry in ASSEMBLER	Entry in COBOL	Flashing	Intensity	Underscore/reverse	Logical end of line	Unprotected field	Protected field	Clear screen	Display on 9752	2nd character set	Standard character set	Numeric input	Video suppression	Delete character	Cursor to start of next line	Position cursor at start of line n
prefix SO	SECOND-CHAR-SET									x1)						
prefix SI	NORMAL-CHAR-SET										x1)					
prefix NL	NEW-LINE		RI		x				YE		x1)				x	
prefix NP	NEW-PAGE		RI					x	YE							
prefix EM1	EMPH-LAYOUT1	x	RI						YE							
prefix EM2	EMPH-LAYOUT2		NI	x					WH							
prefix EM3	EMPH-LAYOUT3		NI						GR							
prefix EM4	EMPH-LAYOUT4		NI	x					RE							
prefix NOR	NORMAL-LAYOUT		RI						YE							
prefix SPA*	START-PROT-AREA		RI				x		YE							
prefix EPA*	END-PROT-AREA		NI			x			GR							
prefix NUM	START-NUM-DATA		NI			x			GR			x				
prefix DAR	DARK-LAYOUT												x			
prefix DEL	DELETE-CHAR													x		
prefix VPA	CONT-LINE															x

Table 7-8: Logical control characters in line mode

7

7.5.3 Logical Support in Format Mode

In format mode, the programmer can employ the Format Handling System FHS, which is incorporated in UTM [28].

The programmer then has at his disposal all the data display terminal functions supported by FHS (see section 7.3 and [28], [30]).

If required, however, other, user-written formatting routines may also be used (see 'Physical Interface' below).

Use of FHS-D

UTM supports decentralized formatting with FHS-D.

Once all the relevant conditions for FHS-D have been met (see [28]) and the formats have been generated, <u>decentralized</u> formatting is peformed automatically.

Partial Formatting

Partial formatting can be used under UTM. Partial formats are formats for which a start line number was defined during format generation with IFG [22].

These formats then occupy only a specified number of lines on the screen.

<u>Note</u>

> Partial formatting is currently not supported by the decentralized formatting system FHS-D.

For further details on partial formatting, see [28].

KDCFORM Macro

When generating the linkage program KDCROOT, the user can specify the following DDT functions by giving the relevant operands in the macro KDCFORM:

DDT function	KDCFORM macro operand
Transfer modified fields only	ISD = RMOD
Transfer all unprotected fields	ISD = RUNP
Set to 'unmodified' or 'unmarked' those fields modified or marked during last input	PMOD = NO
Fields modified or marked during last input retain this status	PMOD = YES
Read with NIL characters or relevant NIL characters	NILS = YES
Read without NIL characters	NILS = NO
Automatic tabulator	ATAB = YES
No automatic tabulator	ATAB = NO

Table 7-9: Operand entries in the macro KDCFORM

Cursor Positioning

The cursor can be positioned on any field by means of the subroutine call CALL "KDCSCUR".

For further information on positioning, see [28].

7.5.4 Physical Interface

The physical interface can be used via the user exit 'Format'.

The user exit activates UTM if the format identifier "-" is used for input and output (entry "-name" in the field KCMF of the KDCS parameter field).

When programming in the physical mode, all DDT control characters described in chapter 4 may be used. Programming is described in [28].

For further information regarding the format of messages issued to the data display terminal, see chapter 4.

Note

In the physical mode, the user himself is responsible for his own formatting program.

This applies particularly to

- restart (creation and modification of the restart area).
- the reaction to error messages.

For further information on the user exit 'Format', see [28].

7.5.5 Badge Reader Support

UTM supports badge readers connected to data display terminals.

The badge reader can be used

- for checking sign-on authorization at KDCSIGN time or
- for data entry using the screen output function KCCARD.

Only one of the two functions can be used for one user ID. If KDCSIGN with a badge (ID card) check has been generated for a user, no further data entry is possible with this badge.

Note

The badge reader is not supported by FHS. A request for data entry with KCCARD is thus only possible in line mode.

For further information on the use of the badge reader, see [28].

7

7.5.6 Printer Support

UTM supports printer output in hardcopy mode and format mode.

The screen output function KCREPR is available for operation in hardcopy mode.

For information about the printers supported in the bypass mode, see [30]. Information on the factors to be considered during programming and generation may be found in [28].

Restrictions

Bypass mode is only possible if the display terminal to which the printer is connected is not connected to an 8170 Cluster Controller (Local) but is supported by PDN.

Only the LA1 function is supported in hardcopy mode.

In central hardcopy mode, only one printer connected to an 8112 Printer Terminal Controller is supported.

Factors to be taken into account when using the 9001, 9003 and 9004 Printers are explained in [26], [12], and [27].

Page

8 PLUS

8.1 GENERAL

With the (Program Key Load and Save) program PLUS the display terminal user can perform the following:

- define the assignment of the P keys in interactive mode,
- save P programs (P formats) in format libraries and
- load the P keys (P registers) of the 9749, 9750 or 9752 Data Display Terminals.

Stored P programs can also be loaded in the P keys by an application program (see 8.4). This is achieved generally using the Format Handling System (FHS) [18].

<u>PLUS offers the following functions:</u>

- definition of P programs in symbolic form with automatic chaining of the free P keys,
- definition of an output message,
- loading the defined P programs at the display terminal,
- documentation of the P programs in symbolic form at the display terminal or on the printer,
- saving of the P programs in an object module library,
- loading of saved P programs.

PLUS can run both in the interactive mode and in batch mode. The P keys can, however, be loaded using PLUS in the interactive mode only with the 9749, 9750 or 9752 Data Display Terminals.

8

8.2 CALLING PLUS

The PLUS program is called up using the EXECUTE command:

/EXEC [$userid.]PLUS

The control statements are then entered. PLUS reads these from the virtual system file SYSDTA.

SYSDTA can be:

- the data display terminal,
- a cataloged SAM or ISAM file,
- the command file SYSCMD.

The assignment can be modified using

- /SYSFILE SYSDTA=
  - before the start of the PLUS program,
  - after the SYS control statement,
  - following detection of the EOF of a cataloged file.

8.3 CONTROL STATEMENTS

Each control statement comprises the operation code and, if necessary, one or more operands. The operation code and operands are separated by at least one blank, the operands themselves by a comma.
The end of the operands is signified by the end of the input record or a blank. Subsequent characters are interpreted as comment and are not processed.
Control statements can be up to 256 characters long. Longer records are truncated.

Operation	Operands	Meaning
Pi␣	'text' cmd[*n] [,...][,Pj] poscmd[:l/c] * - ?	Definition of P key i P program Keep the key free Clear the key Query key program
MSG␣	'text' cmd[*n] [,...] poscmd[:l/c] - ?	Definition of output message Output message Erase output Query output
DEL		Clear all P keys
PUT		Load P programs to the display terminal
SAV␣	mapname[,libname]	Store the P programs in the OML
GET␣	mapname[,libname]	Load the P programs from the OML
DOC␣	 N[ONE] O[UT] L[ST] B[OTH]	Modify documentation medium No documentation Documentation on SYSOUT Documentation on SYSLST Documentation on SYSOUT+SYSLST
SYS		Transition to system mode
END		End of the control statements

Table 8-1 Control statement format

8.3.1 Processing of Control Statements

The operation code and operands undergo a validity check from left to right.
If there is already an entry for the specified P key, the statement is displayed, in interactive mode, together with the existing entry and the user is asked whether or not it should be overwritten. Accordingly, the existing entry is left alone or the new statement is processed.
In batch mode there is no query and the new statement is processed immediately.
If the statement contains an error, it is logged with an appropriate error message on SYSOUT (in interactive mode on the display terminal). Flashing indicates the position of the error. An errored statement is not entered in the PLUS buffer.
If the statement is valid, the corresponding device code is buffered.

Note

In the subsequent examples, input text and output text have the following meaning:

Input text: Text that the user (programmer) must enter in a P key Pi.

Output text: Output/effect on display screen after initiation of the P key Pi.

8.3.2 Description of the Control Statements

P

The P statement defines a program or a statement for the P key specified.

Format

Operation	Operands	Meaning
Pi␣	[{'text' cmd[*n] poscmd[:l/c]} [,...][,Pj] * – ?]	Definition of P key i } P program Keep the key free Clear the key Query key program

Pi␣ P key to be programmed
i=decimal number (l i 20)

The specification of a Pi key must be separated from the operand by means of a blank.

'text' The characters enclosed in quotes are to be stored 1:1 in the P key Pi.
Any single quotes to be stored must be specified twice.

<u>Example</u>

Characters within quotes are to be entered 1:1 in the P key Pi:

Input text: 'PASSWORD C''XYZ'''

Output text: PASSWORD C'XYZ'

cmd Mnemonic designation for a control character sequence to be stored in the P key Pi (in lowercase or uppercase letters).

<u>Example</u>

Data is to be transmitted from start of line to cursor:

Input text: EM,SZA,DUE

*n Repetition factor.
n=decimal number (1 n 99)

The preceding control character sequence is to be copied into the P key Pi n times.

Example

The cursor is to be shifted 8 positions to the right:

Input text: SMR*8

poscmd — Mnemonic for a control character sequence which contains an absolute position specification (POS, LVA, WDH) and is to be stored in the P key Pi.

Example

The cursor is to be positioned to the start of screen and then a hard copy initiated.

Input text: POS,LA1

:l/c — Line and column specification for a positioning control char. sequence.
l/c=decimal number (1 l 24 or 1 c 80)

If no lines and columns are specified, position 1/1 ('home') is set.

Example

Blanks are to be written from the cursor to the end of screen:

Input text: WDH:24/80,' '

Pj — The contents of the P key Pj are to be chained to those of the P key Pi.
j=decimal number (i j 20)

Actuation of the Pi key causes, first of all, the P program i and then the P program j to be executed (explicit chaining).

Example

The P6 key is to be chained to the P4 and P5 keys:

Input text in P4 key: 'SYSFILE SYSDTA',P6
Input text in P5 key: 'SYSFILE SYSLST',P6
Input text in P6 key: '=(PRIMARY)'

Output text after key actuation:
- P4: SYSFILE SYSDTA=(PRIMARY)
- P5: SYSFILE SYSLST=(PRIMARY)

* — The Pi key is to be kept free and not used for automatic chaining.

Example

The P17 key is to be kept free.

Input text in P17 key: *

-	The contents of P key Pi are to be erased. The Pi key is not to be used for automatic chaining.

Example

The register contents of the P1 key are to be erased:

Input text in P1 key: -

?	The program edited for P key i is to be displayed on the screen in symbolic form.

Example

The program for the P6 key is to be displayed symbolically:

Input text: P6 ?

Output text: P06:'=(PRIMARY)'

DEL

The DEL instruction clears all keys.

Format

Operation	Operands	Meaning
DEL		Clear all P keys

If no P keys are assigned, subsequent P or MSG statements are processed without any messages being issued. When the P programs are output, the unused keys are cleared.

PUT

The PUT statement is used to output the defined P programs and, if necessary, a message to the terminal (in interactive mode only).

Format

Operation	Operands	Meaning
PUT		Loading of P programs to the display terminal

The entries in the PLUS buffer for supplying the P areas at the terminal are combined to form a physical output message.
If an entry is longer than 16 bytes, the 'free' P keys are chained. If a message cannot be automatically chained, the statement is logged together with an error message on SYSOUT.
When all the entries have been processed, the P areas of the display terminals are loaded.
The statements are then logged on a defined output medium.

SAV

The SAV statement stores the P programs and output messages in an object module library (OML).

Format

Operation	Operands	Meaning
SAV␣	mapname[,libname]	Store P programs in OML

mapname Name under which the P programs are stored (1-8 alphanumeric characters, the first of which must be alphabetic).

libname Name of a BS2000 object module library (OML) in which the P programs are to be stored (1-54 characters).
If no library name is specified, the P programs are stored in the PLUS.MAPLIB file of the user.

If the operands are permitted, processing takes place as for 'PUT'. The output message is not, however, output to the display terminal, but is stored in a background task.

8

GET

The GET statement loads P programs previoulsy stored using SAV (in the interactive mode only).

Format

Operation	Operands	Meaning
GET␣	mapname[,libname]	Load P programs from OML

mapname Name under which the P programs are stored.
If no library name is specified, the P program is loaded from the user's PLUS.MAPLIB file.

libname Name of a BS2000 object module library (OML) in which the P programs are stored.

If the operands are valid, the P programs are loaded from the file into the program area. If the loading is not completed successfully, a system message to this effect appears and processing is aborted. If the loading is accomplished successfully, the P keys on the display terminals are loaded and the P or MSG messages logged (provided that the documentation medium is SYSOUT). The loaded area is then released again.

Note

The GET control statement does not affect the PLUS internal buffer. If the P programs stored are to be modified, the statements recorded on SYSOUT can be implemented again as control statements.

DOC

The DOC statement assigns the medium on which the P or MSG statements in PUT, GET and END are to be recorded.

Format

Operation	Operands	Meaning
DOC␣		Change documentation medium
	N[ONE]	No documentation
	O[UT]	Documentation on SYSOUT
	L[ST]	Documentation on SYSLST
	B[OTH]	Documentation on SYSOUT+SYSLST

NONE No documentation is required

OUT Documentation on SYSOUT is required (On the terminal in interactive mode).
The statements displayed on the terminal may be modified and re-entered as control statements.

LST Documentation on SYSLST is required. If SYSLST is a cataloged file, it can be used again as an input file.

BOTH Both SYSOUT and SYSLST are to be used for documentation.

SYS

The SYS statement effects a changeover to system mode. General BS2000 commands can then be issued (e.g. reassignment of SYSDTA etc.). A return to program mode is effected by means of the /RESUME command, after which input from SYSDTA continues.

Format

Operation	Operands	Meaning
SYS		Transition to system mode

END

The END statement terminates the program.

Format

Operation	Operands	Meaning
END		End of input

If P programs have been defined but not yet processed with PUT, this processing is initiated.
The program is then terminated.

8.3.3 Maximum Possible P Program Lengths

Without automatic chaining: P1-P20, 16 bytes each

With automatic chaining:

P1:	268 bytes	P6:	198 bytes	P11:	133 bytes	P16:	68 bytes
P2:	254 "	P7:	185 "	P12:	120 "	P17:	55 "
P3:	240 "	P8:	172 "	P13:	107 "	P18:	42 "
P4:	226 "	P9:	159 "	P14:	94 "	P19:	29 "
P5:	212 "	P10:	146 "	P15:	81 "	P20:	16 "

Table 8-2 Maximum possible P program lengths

8.4 LOADING OF P PROGRAMS

8.4.1 Loading of P Programs using a DO Procedure

The DO procedure, which is stored as a file, is started by means of the DO command (system command) [24].

EXAMPLE OF A DO PROCEDURE FOR THE LOADING OF P PROGRAMS

The following procedure is, for example, stored under the file name DO.PKEYS.

```
/PROC
/SYSFILE SYSDTA=(SYSCMD)
/EXEC PLUS
P1 'SYSFILE SYSLST',P3
P2 'SYSFILE SYSDTA',P3
P3 '=(PRIMARY)',EM,DUE
P4 'EXEC $COB1',EM,DUE
P5 'DO DO.COBOL',EM,DUE
P6 'EXEC $SYSUPD',EM,DUE
P7 'LOGON MAS,1234',EM,DUE
P8 'O $DIALOG,1/12',EM,DUE
P9 'EXEC $EDOR',EM,DUE
P10 'EXEC $EDT',EM,DUE
P11 'LOGOFF BUT',EM,DUE
P12 'STA L',EM,DUE
P13 'FSTAT',EM,DUE
P14 'EXEC $COBLUR',EM,DUE
MSG SNZ,'****  P keys now loaded  ****'
PUT
END
/SYSFILE SYSDTA=(PRIMARY)
/ENDP
```

Starting the Procedure File with the DO Command

The stored file with the file name DO.PKEYS is called using the DO command.

```
/DO DO.PKEYS
%  P500 LOADING
****  P keys now loaded  ****
/
```

8.4.2 The Loading of P programs by an Application Program

PLUS stores the programs defined - initiated by the control statement SAV mapname[,libname] - in an object module library. FHS (Format Handling System) can be used for this purpose.
FHS is a chargeable product available for the TIAM (timesharing mode), DCAM, UTM and ASMUS (transaction and inquiry processing) access methods and the programming languages ASSEMBLER and COBOL (as of FHS V3).

The P programs stored by PLUS are output as with a 'normal' FHS format. The format name is the save name used during storage and the format library the corresponding library.
The format can be used purely as an output format or as both an input and output format. The input (requested by text in the MSG control statement) cannot be processed by FHS. It can, however, serve as an acknowledgment (with whatever content) for the loading operation.

Further information on the use of FHS in the individual access methods can be obtained from the FHS User's Guide [18].

8.4.3 Loading of the P Programs with UTM

A P program defined by the PLUS program is stored using the SAV statement

SAV mapname,UTM-format-library

as a FHS format in the corresponding UTM format call library.
The P keys are loaded by means of an MPUT (in a UTM program unit); in which KCMF=*mapname.
Input formatting is carried out after the very next input at the display terminal. This leads to a formatting error in the case of P key formats. Therefore, the program following the loading operation is not allowed to contain an MGET call (otherwise program abortion with 70Z/FR04 and dump).

<u>Example</u>

```
                                      PROG1
                                     +-----------------------------+
------>  TAC for PROG1               |  INIT                       |
                                     | [MGET]                      |
                                     |                             |
                                     |  MPUT   KCOM = NE           |
                                     |         KCMF = <*mapname>   |
                                     |                             |
                                     |  PEND   KCOM = RE/KP        |
<------  Loading of P keys           |         KCRN = <TAC for PROG2>|
                                     +-----------------------------+
------>  Any input                                .
                                                  .
                                      PROG2       .
                                     +-----------------------------+
                                     |  INIT                       |
                                     |                             |
                                     |  MPUT                       |
                                     |                             |
                                     |  PEND   KCOM = FI           |
<------                              +-----------------------------+
```

8.4.4 Loading P Programs without FHS

ASSEMBLER application programs can process stored P programs even without FHS. The system macro

LINK ENTRY=mapname,LIBNAME=libname

loads the area shown below into the program area. After a successful loading operation the register 1 references the start of the area. The start addresses incremented by the respective displacement then have to be entered in the parameter range of the output macros WROUT or WRTRD.

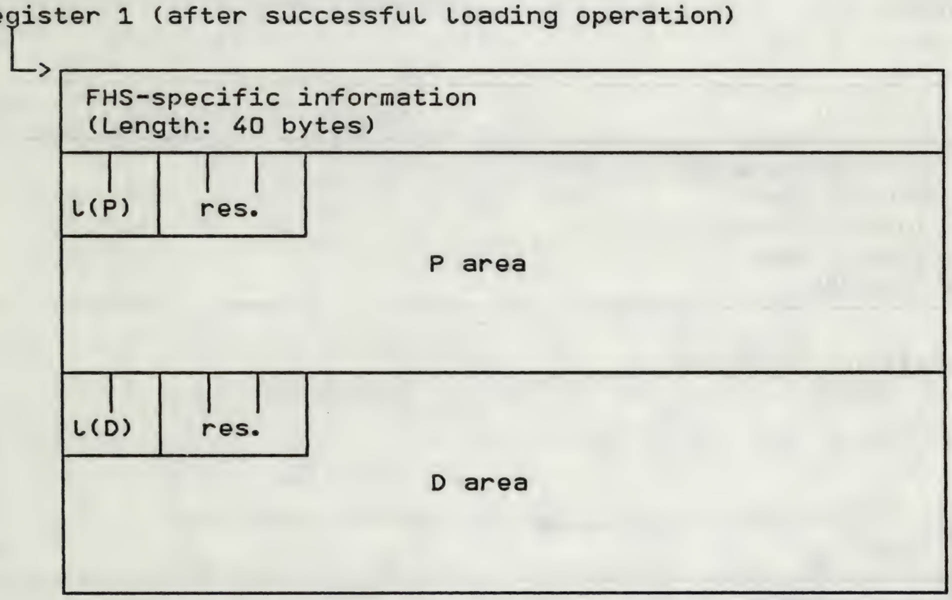
The P area can then be output in physical mode
(in accordance with WROUT/WRTRD ...   ,MODE=PHYS,OHDR=N),

and the D area output in line mode
(in accordance with WROUT/WRTRD ...   ,MODE=LINE).

The loaded area can then be released again by the system macro UNLOD:

<u>Format of the P program area</u>

Register 1 (after successful loading operation)



Area length l(P) or l(D): 2 bytes
Reserved: 3 bytes

The **P area** contains text and device control characters for loading the P programs.
The P programs are documented in the **D area.**

8.5 DEVICE FUNCTION MNEMONICS

Cursor Positioning Functions

Symbol	Length	Meaning cmd
SBA	**	Cursor to home position (start of screen)
SDZ	**	Cursor to start of line above
SML	**	Cursor one position to the left
SMO	**	Cursor up one line
SMR	**	Cursor one position to the right
SMU	**	Cursor down one line
SNZ	**	Cursor to next line
SZA	**	Cursor to start of line
TAL	**	Tabulator to the left
TAR	**	Tabulator to the right

Table 8-3 Cursor positioning functions

Text Shifting Functions

Symbol	Length	Meaning cmd
AFG	**	Delete character
AFZ	**	Delete line
EFG	**	Insert character
EFZ	**	Insert line
RU	**	Roll up

Table 8-4 Text shifting functions

Erasing Functions

Symbol	Length	Meaning cmd
LSP	**	Erase buffer
LVD	**	Erase variable data
LZF	**	Erase to end of line/field
(LVA)		(see positioning device functions)

Table 8-5 Erasing functions

Special Functions

Symbol	Length	Meaning cmd
AKA	**	Audible alarm
AM	***	Start marker
BRS	**	Reset BEL indicator
DEL	*	Smudge character
EM	*	End marker
FAZ	**	Reset field modifications
LZE	**	Logical end of line
MAR	**	Mark field
PAR	**	Modify operand area FD
RS	**	Reset
SP	*	Space
SUB	*	Substitute character
VA	**	Clear connection

Table 8-6 Special functions

Positioning Device Functions

Symbol	Length	Meaning poscmd
LVA:l/c	*****	Erase variable data from cursor up to before position line "l", column "c"
POS:l/c	****	Cursor to position line "l", column "c"
WDH:l/c	*****	Repeat subsequent character from the cursor up to position line "l", column "c"

Table 8-7 Positioning device functions

8.6 GENERAL MESSAGES

MAP SAVING INITIALIZED
Response required: No
Meaning
The ENTER file for saving the P programs has been initialized and opened.

EXISTING: <old p-program> *** OVERWRITE (Y/N)?
Response required: Yes
Meaning
A program has already been defined for a P key.
The terminal user is asked whether it is to be overwritten or not.

ERRORS DETECTED: CONTINUE?
Response required: Yes
Meaning
Errors occurred during processing.
Terminal user is asked if the function responsible should nevertheless be executed.

8.7 ERROR MESSAGES

*** UNKNOWN OPERATION
Meaning
Invalid operation code.
Action
Repeat input with correct code.

*** P-NUMBER OUTSIDE RANGE 1-20
Meaning
Invalid P statement: P key Pi not available.
Action
Repeat input with correct P key number.

*** TO MUCH OPERANDS
Meaning
Too many operands were specified for the defined P key (buffer overflow).
Action
Repeat input with fewer operands or for a lower P key.

*** UNKNOWN OPERAND
Meaning
Invalid mnemonic for an operand.
Action
Repeat input with correct operand.

*** OPERANDS IGNORED AFTER CHAINED KEY
Meaning
A P operand (explicitly chained P key) is not the last operand of a P statement: subsequent operands are ignored.
Action
Specify operands for a chaining key in a separate P statement.

*** DELIMITER MISSING
Meaning
A mandatory delimiter (" ,*:/") is missing from the control statement.
Action
Correct the input.

*** NO KEY TO CHAIN AVAILABLE FOR <Pi>
Meaning
P program too long for the specified P key; no P key free for automatic chaining.

*** KEY TO CHAIN NOT GREATER ACTUAL KEY N.1
Meaning
P key number is too small for explicit chaining.
Action
Repeat input with another chaining key.

*** REPETITION FACTOR OUTSIDE RANGE 1-256
Meaning
Invalid repetition factor.
Action
Repeat input with correct repetition factor.

*** INPUT TOO LONG: TRUNCATED

Meaning

Only the first 256 characters input are interpreted as a control statement. Others are ignored.

*** LINE NUMBER OUTSIDE RANGE 1-24

Meaning

Invalid line specification in a positioning statement.

Action

Repeat input with correct line specification.

*** COLUMN NUMBER OUTSIDE RANGE 1-80

Meaning

Invalid column specification in a positioning statement.

Action

Repeat input with correct column specification.

*** LOADING P-KEYS NOT POSSIBLE ON THIS DEVICE

Meaning

The terminal is not a 9749/9750/9752 Data Display Terminal, therefore no attempt is made to load the P keys.

No action.

*** MAPNAME MISSING

Meaning

The save name is not specified.

Action

Repeat input with correct save name.

*** MAPNAME UNALLOWED

Meaning

The save name is too long or contains an illegal character.

Action

Repeat input with correct save name.

Page

9 DATA DISPLAY TERMINAL FOR OPERATION WITH BERMUDA

9.1 SUMMARY DESCRIPTION

BERMUDA is a program package offering a mask support service for display terminal users. It is a component of the PDN operating system used in Siemens TRANSDATA 9.600 systems, which comprise terminal computers and terminals. BERMUDA can be employed on data display terminals connected to terminal computers of the TRANSDATA 960 system.

BERMUDA is intended to be used for the implementation of preprocessing functions typical of decentralized computers in distributed processing environments.

BERMUDA brings about a substantial improvement in the man-machine dialog, as well as reducing the load on the host computer, by providing

- support for the data display terminal via format-controlled input/output,
- a data check accompanying input in the case of field-by-field transfer, and immediate system reaction to input errors on the part of the user,
- data storage in the (decentralized) terminal computer, and transfer of data relevant to processing (net data) to the host at the most favorable moment from the point of view of the processing task.

All the other facilities offered by TRANSDATA PDN can also be used in conjunction with BERMUDA (see [13]).

This chapter provides an overview of the programming of 9750 and 9752 Data Display Terminals, with particular reference to the functions of these terminals when used in conjunction with BERMUDA.

Keyboard

A special keyboard is required when employing BERMUDA with 9750 and 9752 Data Display Terminals.
A detailed description of the BERMUDA keyboard may be found in [31].

9.2 OPERATING MODES

BERMUDA supports the field and record modes.

Field Mode

The field mode is selected in character 6 (FST4) of PAR OOL. In field mode BERMUDA offers the data display terminal user only one variable field for data input purposes.
In this variable field the user must (mandatory field) or may (non-mandatory field) enter his data. A check on the data is performed immediately the field has been filled.

A variable field can be preloaded with data before it is output.
Prior to data entry, BERMUDA makes available to the data display terminal the rules regarding input to the variable field. These rules include

- position and length of the variable field,
- the valid input characters,
- the type of display,
- the user prompts,
- additional device control characters.

Each character entered via the data display terminal is immediately checked for validity and correctness of syntax.
Provided that it conforms to the rules, the character entered is then displayed on the screen.

Record Mode

Record mode differs from field mode in that the data entered is checked only when the entire form has been filled in.
This means that in this case a number of variable fields are output, instead of just one as in field mode.

9.3 MESSAGE FORMAT

In principle, the message format for BERMUDA operation is the same as that described in chapter 4.
The only factor requiring particular consideration is the individual formatting of specific BERMUDA parameter ranges as described in this chapter.

9.3.1 Message Prefix for Data Entry via a Data Display Terminal

For data input from a display terminal, the message prefix consists of at least the start-of-message character plus an 8-byte message header (MH = PAR 00E).
If a text ID is also dispatched, the message header is 16 bytes long in total (MH = PAR 00E + PAR 01E).
In the field mode (character 6 (FST4) of PAR 00L = X'42') the message header is 24 bytes in length (MH = PAR 00E + PAR 01E + PAR 02E). For each transmission to the host, the display terminal places the necessary parameter ranges (as specified by character 5 (FST3) of PAR 00L) ahead of the message.

Direction of transmission	SOM	Meaning
Terminal-to-host		Record mode:
	X'48'	Header comprises PAR 00E
	X'50'	Header comprises PAR 00E + PAR 01E
		Field mode:
	X'58'	Header comprises PAR 00E + PAR 01E + PAR 02E

Table 9-1: Message prefix for data input

9.3.2 Functions of the Parameter Ranges

PAR 00E: Device-specific information is entered in this range.

PAR 01E: This parameter range is the extension of PAR 00E and is for entering the header length, specifying the cursor position before transmission is initiated, and for noting error messages.

PAR 02E: This parameter range is reserved for field mode.
Each time data is entered, the display terminal uses it for entries referring to the type of input.

Format of the Parameter Range PAR 00E

Character	Designation	Meaning
1	SAW	Send command
2	PFLNG	Buffer length
3	ZLNG	Line length
4	AUSG	Program revision level
5	ZZ1	Status character 1
6	ZZ2	Status character 2
7	CDS	Send-key code
8	WAR	Queue control character

Table 9-2: Overview of PAR 00E characters

Char.	Design.	Meaning	Coding	Explanation
1	SAW	Send command	Dependent on whether SAW1 or SAW2 is used.	Depending on the transmit key used (DÜ1 or DÜ2), the corresponding send command SAW1 or SAW2 is entered in PAR 00E and executed. The functions SAW1 and SAW2 are specified (via host-to-terminal transmission) in PAR 00L and PAR 01L respectively. After terminal switchon or user initiation of 'LSP' function, SAW=X'00'.
2	PFLNG	Buffer length	X'40'	——
3	ZLNG	Line length	X'30'	——
4	AUSG	Program revision level	X'60'	Program memory: DIALOG Program memory: DIALOG + BERMUDA
			X'67'	Program memory: DIALOG
			X'64'	Program memory: DIALOG
			X'65'	Program memory: DIALOG + BERMUDA with key verification
			X'66'	Program memory: Dialog for X.21
5	ZZ1	Status char. 1 Contains user ID (badge reader and keylock switch) and is an extension of ZZ2.	Bit 7 6 5 4 3 2 1 (bit 7 = 1) Bit 1	=0: no badge reader ID present (badge not inserted) =1: badge reader ID present (badge inserted) 1)
			Bit 2	=0: / =1: Keylock switch 1 2)
			Bit 3	=0: / =1: Keylock switch 2 2)
			Bit 4	=0: / =1: Keylock switch 3 2)
			Bit 5	=0: / =1: Local device with GAD6 3)
			Bit 6	=0: / =1: Local device with GAD7 3)

Char.	Design.	Meaning	Coding	Explanation	
6	ZZ2	Status char. 2	Bit 7 6 5 4 3 2 1; Bit 7: 1		
			Bit 1	=0: / =1: Keyboard	3)
			Bit 2	=0: / =1: Local device with GAD1	3)
			Bit 3	=0: / =1: Local device with GAD2	3)
			Bit 4	=0: / =1: Local device with GAD3	3)
			Bit 5	=0: / =1: Local device with GAD4	3)
			Bit 6	=0: / =1: Local device with GAD5	3)
7	CDS	Send-key code. Code of actuated send key entered in this character	Record mode: See Table 4-33 "Send functions".	Transmission as per Table 4-33 "Send functions."	
			Field mode: X'00'	No meaning	
8	WAR	Queue control character	Arbitrary, but no transmit control characters.	The terminal returns the host-specified WAR, unmodified, to the host.	

1) Bit 1 can be reset by
   - removing the badge
   - issuing a K14 message via the keyboard.

   Transmission of badge reader information to the host is only possible in block mode (record mode).

2) 0: Keylock switch 'OFF'
   1: Keylock switch 'ON'

3) 0: Device inoperable, i.e.
   - disconnected or
   - malfunctioning (device-specific status message)

   1: Device operable

Table 9-3: Description of PAR 00E characters

Format of Parameter Range PAR 01E

Char.	Design.	Meaning	Coding	Explanation
1	TIL	Text ID length	X'40'	No text identification TI
			X'41'	TI = 8 bytes (PAR 01D)
			X'42'	TI = 16 bytes (PAR 01D + PAR 02D)
			X'43'	TI = 24 bytes (PAR 01D + PAR 02D + PAR 03D)
2	--	--	X'00'	Reserved
3	ZLA	Line address	Dependent on FST3 of PAR 00L (bit 2)	Characters 3, 4 and 5 indicate cursor position before activation of a DÜ, K or F key.
4	SPA	Column address		
5	SAD	Page address	X'30'	
6	FEM	Error message	See Table 4-7	For error messages
7	--	--	X'00'	Reserved
8	--	--	X'00'	Reserved

Table 9-4: Description of PAR 01E characters

Format of Parameter Range PAR 02E

Char.	Design.	Meaning	Coding	Explanation
1	ZZ3	Status character 3	X'40'	VALID not actuated
			X'41'	VALID actuated
2	STA1	Statistics 1	X'40' - X'7F'	Number of corrections on key verification
3	STA2	Statistics 2	X'40' - X'7F'	Number of strokes in the field, represented in binary, with STA3 containing the low-order byte
4	STA3	Statistics 3		
5	STA4	Statistics 4	X'40' - X'7F'	Number of flashers in the field
6	--	--	X'40'	No meaning
7	CDF1	Function code	X'40' - X'7F'	In the case of a 2-character string, CDF1 20, CDF1 contains the code of the device function initiating transmission. In the case of a 3-character string, CDF1=20, CDF2 contains the code of the device function initiating transmission.
8	CDF2	Function code	X'40' - X'7F'	

Table 9-5: Description of PAR 02E characters

9.3.2.1 Message Section with Parameter Entries

The format of parameter entries - e.g. coding of parameter ranges and parameter sections - is as described in 4.2.1 and the subsequent sections. The user need only take note of those entries in PAR 00L which differ, plus the additional parameter range PAR 02L for field mode operation. These differences from section 4.2.1 are described below.

Format of Parameter Range PAR 00L

Char.	Design.	Meaning	Coding	Explanation
1	SAW1 1)	Send command 1	X'47'	Transmission of current field only, i.e. that which contains the cursor.
2	FST1	Function control character 1	X'49'	Functions EFZ and AFZ via keyboard disabled. Protected fields can be reached with cursor functions.
3	FST2	Function control character 2	X'79'	From start to end of the screen MOD and flasher bits in unprotected fields are erased. If LFZ is entered, the MOD and flasher bits remain unchanged.
4	GEF1	Device function character 1	X'00'	No meaning. All device functions controlled via PAR 00D.
5	FST3	Function control character 3	X'5C'	Dialog mode activated. Cursor remains stationary during input. For input, the message header consists of PAR 00E + PAR 01E + PAR 02E.
6	FST4	Function control character 4	X'42'	Field mode activated. For input, the message consists of PAR 00E + PAR 01E + PAR 02E.
7	GEF2	Device function character 2	X'00'	No meaning. All device functions controlled via PAR 00D.
8	WAR	Queue control character	X'44'	Is passed from the host to the DDT in unmodified form (character 8 of PAR 00E).

1) In field mode - with and without key verification - SAW1 = X'47' should be used. Each time transmission is initiated, the field containing the cursor is defined as the current field and passed to the host. The host must not enter AM and LZE in the current field. These characters cannot be entered via the keyboard in field mode.

Table 9-6: Description of PAR 00L characters

Format of Parameter Range PAR 02L

Char.	Design.	Meaning	Coding	Explanation
1	——	——	X'40'	reserved
2	FVG1	Preset field value 1	Bit 7 6 5 4 3 2 1; Bit 7: 1	
			Bit 1	=0: Blank and uppercase letters permitted =1: Blank and uppercase letters except for A, B, C, D, E and F not permitted. The uppercase letters A - F are also used as hexadecimal numbers; they are only blocked if hex numbers are also blocked, i.e. bit 6=1 too.
			Bit 2	=0: decimal numbers permitted =1: decimal numbers not permitted; effective only when bit 6=1
			Bit 3	=0: Class 1 special characters permitted, these are: ! " ä & ' (- = ? / ; : * @ $ =1: not permitted
			Bit 4	=0: period and comma permitted =1: not permitted
			Bit 5	=0: plus and minus permitted =1: not permitted
			Bit 6	=0: hex numbers (0 to F) permitted =1: not permitted

Char.	Design.	Meaning	Coding	Explanation
3	FVG2	Preset field value 2	Bit 7 6 5 4 3 2 1 1 0	
			Bit 1	=0: lowercase letters and blank permitted =1: not permitted
			Bit 2	=0: fill field with blanks =1: fill field with null characters
			Bit 4	=0: non-mandatory field =1: mandatory field
			Bit 5	=0: not total-fill field =1: total-fill integrity field
			Bit 6	=0: Class II special characters permitted, these are: lowercase and uppercase umlauts and ß =1: not permitted
4	LH	Field length	X'30' - X'39'	Hundreds
5	LZ	Field length	X'30' - X'39'	Tens
6	LE	Field length	X'30' - X'39'	Units
7	FKZ	Function identifier	Bit 7 6 5 4 3 2 1 1 0 0 0 0	
			Bit 1	=0: no key verification =0: key verification
			Bit 2	=0: KOR permitted =1: KOR not permitted
8	——	——	X'40'	Reserved

Table 9-7: Description of PAR 02L characters

9.4 INDICATOR LINE

In the record and field modes, the data display terminal provides the user with a status indicator line, which occupies the 25th line of the screen. Depending on character 5 (AZL) of PAR 00D, this can function either as a device status line or as a system line.

In the data display terminal's initial state (at switchon), the device status line is displayed automatically.
The device status line serves to display operating states; in the field mode it also indicates any syntax errors [31].

In BERMUDA operation the system line is designated the BERMUDA indicator line, and is used for the output of error texts [31].

Further information on the message format for system message output may be found in section 4.3.

9.5 FUNCTION KEYS

The function keys of the BERMUDA keyboard are divided into

- system function keys,
- user function keys,
- short code keys.

The system function keys are divided into

- correction functions,
- positioning functions,
- termination functions,
- operating aids.

For further details on keyboard functions, see section 6.2.

9.6 KEY VERIFICATION

In the case of variable fields, whose contents cannot be checked automatically, checks for input errors can be made by repeating the data input and comparing it with the first input.
This operating mode can be selected in character 7 (FKZ) of PAR 02L, bit 1=1.
The characters to be checked appear as smudge characters on the screen. If the same character as that first entered is input, the cursor moves on to the next smudge character or alternatively to the next field if the end of the field has been reached.

If the character entered does not tally with the original input, the flasher is set. A letter 'P' is displayed in the field at the left-hand end of the BERMUDA indicator line (area for error texts) to denote an error.

For further information on key verification, see [13] and [15].

10 9750-5 DATA DISPLAY TERMINAL WITH X.21 INTERFACE

This chapter is to be supplied as part of a subsequent revision. Details of publication will be given in the publication announcement ("Druckschriften-Neuerscheinung").

10

11 9750-62 DATA DISPLAY TERMINAL

This chapter is to be supplied as part of a subsequent revision. Details of publication will be given in the publication announcement ("Druckschriften-Neuerscheinung").

A APPENDIX

A.1 TABLES

Device functions

ISO code

Two-character sequence

	0	1	2	3	4	5	6	7	
0				AKA	SMR	PAR	RU	BRS	0
1					SML	LA1	P2	LZE	1
2					SMO	LA2	SBA		2
3					SMU	K1	P1		3
4					SNZ	K2	LSP		4
5					SZA	K3	LVD		5
6					TAR	K4	DÜ1		6
7					TAL	K5			7
8					EFG		P3		8
9		EM			AFG		P4		9
A			P5	K14	AFZ		P		A
B		ESC	P6	K13	EFZ	F1	SDZ		B
C		IS4		K12	LZF	F2	MAR		C
D		IS3		K11	K6	F3	FAZ		D
E		IS2		K10	K7	F4	RS		E
F				K9	K8	F5	VA		F
	0	1	2	3	4	5	6	7	

Example
Cursor to start of screen (SBA): ESC-GEF = X'1B62' (ISO-7-Bit-Code).

IS 2 =
IS 3 =
IS 4 = pos. command (blz 4-28)

A

Three-character sequence

	0	1	2	3	4	5	6	7	
0			SP		P7	AM	FON		0
1					P8	DÜ2	PAK		1
2					P9	SS	WDH		2
3					P10		LVA		3
4					P11				4
5					P12				5
6					P13				6
7					P14				7
8					P15				8
9					P16				9
A					P17				A
B		ESC			P18	LA3			B
C					P19	LA4			C
D					P20	LA5			D
E						LA6			E
F						LA7			F
	0	1	2	3	4	5	6	7	

Example

Initiation of the P 20 function: ESC-Space-GEF = X'1B204D' (ISO-7-Bit-Code).

POSITIONING COMMANDS

Line address

Version A			Version B		
	Coding			Coding	
Line	EBCDIC	ISO-7	Line	EBCDIC	ISO-7
1	07	7F	1	F7	37
2	FF	7E	2	F6	36
3	FD	7D	3	F5	35
4	4F	7C	4	F4	34
5	FB	7B	5	F3	33
6	A9	7A	6	F2	32
7	A8	79	7	F1	31
8	A7	78	8	F0	30
9	A6	77	9	61	2F
10	A5	76	10	4B	2E
11	A4	75	11	60	2D
12	A3	74	12	6B	2C
13	A2	73	13	4E	2B
14	99	72	14	5C	2A
15	98	71	15	5D	29
16	97	70	16	4D	28
17	96	6F	17	7D	27
18	95	6E	18	50	26
19	94	6D	19	6C	25
20	93	6C	20	5B	24
21	92	6B	21	7B	23
22	91	6A	22	7F	22
23	89	69	23	5A	21
24	88	68	24	40	20

A

Column address

Column	Coding		Column	Coding		Column	Coding		Column	Coding	
	EBCDIC	ISO-7		EBCDIC	ISO-7		EBCDIC	ISO-7		EBCDIC	ISO-7
1	07	7F	21	92	6B	41	E6	57	61	C3	43
2	FF	7E	22	91	6A	42	E5	56	62	C2	42
3	FD	7D	23	89	69	43	E4	55	63	C1	41
4	4F	7C	24	88	68	44	E3	54	64	7C	40
5	FB	7B	25	87	67	45	E2	53	65	6F	3F
6	A9	7A	26	86	66	46	D9	52	66	6E	3E
7	A8	79	27	85	65	47	D8	51	67	7E	3D
8	A7	78	28	84	64	48	D7	50	68	4C	3C
9	A6	77	29	83	63	49	D6	4F	69	5E	3B
10	A5	76	30	82	62	50	D5	4E	70	7A	3A
11	A4	75	31	81	61	51	D4	4D	71	F9	39
12	A3	74	32	4A	60	52	D3	4C	72	F8	38
13	A2	73	33	6D	5F	53	D2	4B	73	F7	37
14	99	72	34	6A	5E	54	D1	4A	74	F6	36
15	98	71	35	BD	5D	55	C9	49	75	F5	35
16	97	70	36	BC	5C	56	C8	48	76	F4	34
17	96	6F	37	BB	5B	57	C7	47	77	F3	33
18	95	6E	38	E9	5A	58	C6	46	78	F2	32
19	94	6D	39	E8	59	59	C5	45	79	F1	31
20	93	6C	40	E7	58	60	C4	44	80	F0	30

Page address

<u>Only</u> the page address SAD = X'30' is used for the 9749, 9750 and 9752 Data Display Terminals.

Distance	Coding	
	EBCDIC	ISO-7
0	F0	30

FIELD HANDLING CHARACTERS

Field attribute	Coding EBCDIC	ISO-7	Field attribute	Coding EBCDIC	ISO-7
V,NM,NP,A	7C	40	PS,NM,NP,A	4A	60
V,NM,NP,N	C2	42	PS,NM,NP,N	82	62
V,NM,PF,A	D7	50	PS,NM,PF,A	97	70
V,NM,PF,N	D9	52	PS,NM,PF,N	99	72
V,MA,NP,A	C8	48	PS,MA,NP,A	88	68
V,MA,NP,N	D1	4A	PS,MA,NP,N	91	6A
V,MA,PF,A	E7	58	PS,MA,PF,A	A7	78
V,MA,PF,N	E9	5A	PS,MA,PF,N	A9	7A
P,NM,NP,A	C1	41	PA,NM,NP,A	81	61
P,NM,NP,N	C3	43	PA,NM,NP,N	83	63
P,NM,PF,A	D8	51	PA,NM,PF,A	98	71
P,NM,PF,N	E2	53	PA,NM,PF,N	A2	73
P,MA,NP,A	C9	49	PA,MA,NP,A 1)	89	69
P,MA,NP,N	D2	4B	PA,MA,NP,N	92	6B
P,MA,PF,A	E8	59	PA,MA,PF,A	A8	79
P,MA,PF,N	BB	5B	PA,MA,PF,N	FB	7B

1) Combinations with PA are not meaningful

Key to the Abbreviations

V Variable field
P Protected field, not returnable
PS Protected field, returnable
PA Protected field, only field addresses are returnable
A Alphanumeric field
N Numeric field
NM Non-markable field
MA Markable field
NP Non-printable field
PF Printable field

DISPLAY CONTROL CHARACTERS

Display attribute	Coding EBCDIC	ISO-7
NI, SF, NF	7C	40
NI, SF, FL	C1	41
NI, US, NF	C2	42
NI, US, FL	C3	43
RI, SF, NF	C8	48
RI, SF, FL	C9	49
RI, US, NF	D1	4A
RI, US, FL	D2	4B
BL 1)	C4	44

1) Attribute combinations with BL (blanking) are not meaningful

Key to the Abbreviations

NF Non-flashing
FL Flashing
SF Standard Font
US Underscore
BL Blanking
NI Normal intensity
RI Reduced intensity

CODE TABLES

EBCDI Code (international version)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0	NUL	DLE			SP	&	-									0	0
1	SOH	DC1					/		a	j			A	J		1	1
2	STX	DC2		SYN					b	k	s		B	K	S	2	2
3	ETX	DC3							c	l	t		C	L	T	3	3
4									d	m	u		D	M	U	4	4
5	HT		LF						e	n	v		E	N	V	5	5
6		BS	ETB						f	o	w		F	O	W	6	6
7	DEL		ESC	EOT					g	p	x		G	P	X	7	7
8		CAN							h	q	y		H	Q	Y	8	8
9		EM							i	r	z		I	R	Z	9	9
A					`	!	^	:									A
B	VT				.	¤	,	#				[				{	B
C	FF	IS4		DC4	<	*	%	@				\					C
D	CR	IS3	ENQ	NAK	(	)	_	'				]				}	D
E	SO	IS2	ACK		+	;	>	=									E
F	SI	IS1	BEL	SUB	|		?	"								¬	F
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	

Example

The letter 'A' has the coding 'C1' (EBCDIC).

EBCDI Code (German version)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0	NUL	DLE			SP	&	-									0	0
1	SOH	DC1					/		a	j			A	J		1	1
2	STX	DC2		SYN					b	k	s		B	K	S	2	2
3	ETX	DC3							c	l	t		C	L	T	3	3
4									d	m	u		D	M	U	4	4
5	HT		LF						e	n	v		E	N	V	5	5
6		BS	ETB						f	o	w		F	O	W	6	6
7	DEL		ESC	EOT					g	p	x		G	P	X	7	7
8		CAN							h	q	y		H	Q	Y	8	8
9		EM							i	r	z		I	R	Z	9	9
A					`	!	^	:									A
B	VT				.	$	,	#				Ä				ä	B
C	FF	IS4		DC4	<	*	%	§				Ö					C
D	CR	IS3	ENQ	NAK	(	)	_	'				Ü				ü	D
E	SO	IS2	ACK		+	;	>	=									E
F	SI	IS1	BEL	SUB	ö		?	"								ß	F
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	

Example

The letter 'A' has the coding 'C1' (EBCDIC).

ISO 7 bit code (international version)

	0	1	2	3	4	5	6	7	
0	NUL	DLE	SP	0	@	P		p	0
1	SOH	DC1	!	1	A	Q	a	q	1
2	STX	DC2	"	2	B	R	b	r	2
3	ETX	DC3	#	3	C	S	c	s	3
4	EOT	DC4	¤	4	D	T	d	t	4
5	ENQ	NAK	%	5	E	U	e	u	5
6	ACK	SYN	&	6	F	V	f	v	6
7	BEL	ETB	'	7	G	W	g	w	7
8	BS	CAN	(	8	H	X	h	x	8
9	HT	EM	)	9	I	Y	i	y	9
A	LF	SUB	*	:	J	Z	j	z	A
B	VT	ESC	+	;	K	[	k	{	B
C	FF	IS4	,	<	L	\	l	|	C
D	CR	IS3	-	=	M	]	m	}	D
E	SO	IS2	.	>	N	^	n	¬	E
F	SI	IS1	/	?	O	_	o	DEL	F
	0	1	2	3	4	5	6	7	

Example

The letter 'A' has the coding '41' (ISO 7 bit code).

ISO 7 bit code (German version)

	0	1	2	3	4	5	6	7	
0	NUL	DLE	SP	0	§	P		p	0
1	SOH	DC1	!	1	A	Q	a	q	1
2	STX	DC2	"	2	B	R	b	r	2
3	ETX	DC3	#	3	C	S	c	s	3
4	EOT	DC4	$	4	D	T	d	t	4
5	ENQ	NAK	%	5	E	U	e	u	5
6	ACK	SYN	&	6	F	V	f	v	6
7	BEL	ETB	'	7	G	W	g	w	7
8	BS	CAN	(	8	H	X	h	x	8
9	HT	EM	)	9	I	Y	i	y	9
A	LF	SUB	*	:	J	Z	j	z	A
B	VT	ESC	+	;	K	Ä	k	ä	B
C	FF	IS4	,	<	L	Ö	l	ö	C
D	CR	IS3	-	=	M	Ü	m	ü	D
E	SO	IS2	.	>	N	^	n	¬	E
F	SI	IS1	/	?	O	_	o	DEL	F
	0	1	2	3	4	5	6	7	

<u>Example</u>

The letter 'A' has the coding '41' (ISO 7-bit code).

Meaning of the Control Characters in the Code Tables

NUL	Null (fill) character	DLE	Data link escape character (for switching to transparent mode)
SOH	Start of header		
STX	Start of Text	DC1	Device control character 1
ETX	End of Text	DC2	Device control character 2
EOT	End of transmission	DC3	Device control character 3
ENQ	Query	DC4	Device control character 4
ACK	Positive acknowledgment	NAK	Negative acknowledgment
BEL	Visual alarm (combination with audible alarm possible)	SYN	Synchronous idle character
		ETB	End of transmission block
BS	Backspace	CAN	Cancel character
HT	Horizontal tabulator character	EM	End marker
LF	Line feed	SUB	Substitution
VT	Vertical tabulator character	ESC	Code switchover
FF	Form feed	IS4	Information separator character 4
CR	Carriage return	IS3	Information separator character 3
SO	Shift-out	IS2	Information separator character 2
SI	Shift-in	IS1	Information separator character 1
SP	SPACE	DEL	Delete

CODE CONVERSION TABLES

EBCDIC --> ISO 7-bit code

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0	00	10			20	26	2D									30	0
1	01	11					2F		61	6A			41	4A		31	1
2	02	12		16					62	6B	73		42	4B	53	32	2
3	03	13							63	6C	74		43	4C	54	33	3
4									64	6D	75		44	4D	55	34	4
5	09		0A						65	6E	76		45	4E	56	35	5
6		08	17						66	6F	77		46	4F	57	36	6
7	7F		1B	04					67	70	78		47	50	58	37	7
8		18							68	71	79		48	51	59	38	8
9		19							69	72	7A		49	52	5A	39	9
A					60	21	5E	3A									A
B	0B				2E	24	2C	23				5B				7B	B
C	0C	1C		14	3C	2A	25	40				5C					C
D	0D	1D	05	15	28	29	5F	27				5D				7D	D
E	0E	1E	06		2B	3B	3E	3D									E
F	0F	1F	07	1A	7C		3F	22								7E	F
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	

Example

Conversion: 'C1' (EBCDIC) ≙ '41' (ISO 7-bit code).

ISO 7-bit Code --> EBCDIC

	0	1	2	3	4	5	6	7	
0	00	10	40	F0	7C	D7	4A	97	0
1	01	11	5A	F1	C1	D8	81	98	1
2	02	12	7F	F2	C2	D9	82	99	2
3	03	13	7B	F3	C3	E2	83	A2	3
4	37	3C	5B	F4	C4	E3	84	A3	4
5	2D	3D	6C	F5	C5	E4	85	A4	5
6	2E	32	50	F6	C6	E5	86	A5	6
7	2F	26	7D	F7	C7	E6	87	A6	7
8	16	18	4D	F8	C8	E7	88	A7	8
9	05	19	5D	F9	C9	E8	89	A8	9
A	25	3F	5C	7A	D1	E9	91	A9	A
B	0B	27	4E	5E	D2	BB	92	FB	B
C	0C	1C	6B	4C	D3	BC	93	4F	C
D	0D	1D	60	7E	D4	BD	94	FD	D
E	0E	1E	4B	6E	D5	6A	95	FF	E
F	0F	1F	61	6F	D6	6D	96	07	F
	0	1	2	3	4	5	6	7	

Example

Conversion: '41' (ISO 7-bit code) ≙ 'C1' (EBCDIC).

A

A.2 GLOSSARY

Application program (BERMUDA)	Processing specifications for controlling an application. The application program controls the action. The data input by the user in response to a format-controlled screen layout is analyzed and, if required, passed on for further processing in accordance with the scheme defined in the application program.
Applications programming language (APS)	Application-oriented programming language of the TRANSDATA 960 System which is used for the formulation of application programs.
Attribute	Characteristic of a field and is specified by the programmer.
Dark field	Data in a dark field is blanked (not visible on the screen).
Data communication	Language component used for message exchange between communication partners in the TRANSDATA network (between one application and another).
Device control characters	Physical control characters which initiate certain device functions at the terminal.
Duplication	The current field is filled with the data of the duplicate record.
Error message	Message to the application program when a current message cannot be processed by logical input support (incorrect message format, transmission error, logical input support not available).
Flasher	Screen contents flash as a result of an incorrect input by the terminal user. At the same time an audible alarm sounds and the keyboard is is locked.

A

Format terminal	A type of virtual terminal. The data structure is formed by fields with differing characteristics.
Function code	Symbolic name of a device function.
Input matching	A component of MULTIKOS for providing logical input support.
Integrity field (= total-fill field)	The terminal user must fill in this field completely before he can leave it.
Keyboard functions	Terminal response to the pressing of a function key.
Line terminal	The line terminal is implemented in the system by the VTSU (virtual terminal support) module. The programmer has only to arrange the data into logical lines.
Local printer	Printer connected to a data display terminal as a local device that must be addressed as a component of the data display terminal (YSDEV PRNT).
Logical control character	Logical control characters within the message initiate, after conversion by the VTSU, functions in the terminal.
Mandatory field	The terminal user has to enter at least one character in this field.
MASK	Screen language used to define formats.
Terminal component	A terminal comprises one or more input/output components e.g. keyboard, screen, printer etc.
Transmit instruction	Tells the terminal what data is to be transmitted as the input message.
TRINIDAD	TRANSDATA Integrated Data Entry Services. PDN component that supports the terminal user in acquiring data.
User control character handling	User procedure for processing those control characters to which no system functions are assigned.

User procedure	Program modules written in APS by the user and intended for the character by character, field-by-field and record-by-record processing of input data.
User prompting	Program-controlled support for the terminal user during data input by means of explantory texts or warnings.
Variable field	A defined area of a form in which data is entered.
Virtual terminal	Virtual terminals relieve the programmer of the need for knowledge of message construction. Virtual terminals are user service modules in the TRANSDATA data communication system.

A

A.3 ABBREVIATIONS

ADA	Print command
AD1/2	Address 1/2
AKT	Termination of acquisition
APS	Programming language for defining application programs
ASZ	Display control character
ATAB	Automatic tabulator
AWL	Badge reader
AZL	Status line
BAM	Bit-serial communciation protocol for connection to cluster controllers (bit-serial asynchronous multi-point line protocol)
BAST	BAM interface controller
BE	End of screen
BERMUDA	User Services for Terminal Mask Support
BK	Mailbox facility
CCB	Control character block
CDS	Code of the send key
CFEP	Compact front-end processor
DCAM	Data Communication Access Method
DCE	Data communication equipment
DCM	Data Communication Methods
DDT	Data display terminal
DSPLY	Display screen
DSR	Terminal computer
DSS-E	Standalone data display terminal
DSS-M	Cluster data display terminal
DÜ	Data transmission
DVA	Data processing system
DXC	Data exchange control
EA	Selecting sequence
EBCDIC	Extended Binary Coded Decimal Interchange Code
EINF	Record insertion
EM	End marker
FAZ	Set field separators to initial state
FB	Field mode
FBZ	Field handling character
FEM	Error message
FEP	Front-end processor
FEPR	Remote front-end processor
FHS	Format Handling System
FORM	Format branch
FST	Function control character
FTZ	Field separator
GEF	Device function
ID1/2	Identification characters 1/2
IE	Interface expander
IFEP	Integrated front-end processor
IFG	Interactive Format Generator
ISO	International Organisation for Standardisation
ITC	Integrated terminal controller
KAN	Channel address
LAP1	LA function parameter 1
LAx	Local initiation
LCC	Local cluster controller
LOES	Record erasure
LSV1	Low-speed variant 1 (protocol)
MH	Message header
MOD	Modification
MSC	Medium-speed concentrator

A

MSV1	Medium-speed variant 1 (protocol)
MT	Message text
MULTIKOS	Multi-component terminal support
OCR	Optical character recognition
OML	Object module library
PAG	Parameter entries
PAK	Parameter identifier character
PBH	Parameter handling
PCC	Protocol converter concentrator
PDN	Program System for Teleprocessing and Network Control
PLUS	Program Key Load and Save System
PRNT	Printer
RB1/2	Acknowledgment bytes 1/2
RCC	Remote cluster controller
RPG	Report Program Generator
RS	Reset
SA	Polling
SAD	Page address
SATZ (<--)	Turn over backwards
SATZ (-->)	Turn over forwards
SAW	Send instruction
SB	Record mode
SE	Station header
SM	Cursor
SOM	Start-of-message character
SP	Space
SPA	Column address
SS	Interface
STB	Status byte
STZ	Physical control character
SUCH	Record search
TI	Text identification (ID)
TIAM	Terminal Interactive Access Method
TIL	Text identification length
UNT	Interruption of data acquisition
UTM	Universal Transaction Monitor
VAR	Host computer
WAR	Queue control character
ZLA	Line address
ZW	Change of status
ZZx	Status characters

A.4 LITERATURE

[1] **Datenverarbeitung ohne Grenzen** *)
(Introduction to teleprocessing with TRANSDATA)

[2] **Product Spectrum TRANSDATA**

[3] **Teleprocessing with BS2000**
Summary Description

[4] **Tastaturen 97500-xx** *)
Datensichtstationen
Beschreibung (Description of 97500-xx keyboards)

[5] **8906-2 Interface Expander**
Data Sheet

[6] **8901, 8903 Concentrators**
Data Sheet

[7] **Datensichtstation 9750** *)
Einstellvorschrift (Adjusting Specifications)

[8] TRANSDATA
PDN
Interfaces to Data Communication
User's Guide

[9] TRANSDATA
PDN
APS
Applications Programming Language
User's Guide

[10] TRANSDATA
BS2000
PDN
Administration and Services
of a Data Communication System
User's Guide

[11] TRANSDATA
BS2000
PDN
Generation of a Data Communication System
User's Guide

[12] **9003 Printer**
Programmer's Interface
User's Guide

[13] TRANSDATA
PDN
BERMUDA
User Service for
Terminal Mask Support
Reference Manual

[14] TRANSDATA
BS2000
PDN
COBOL9600
User's Guide

[15] TRANSDATA
PDN
BERMUDA
User Service for
Terminal Mask Support
User's Guide

[16] TRANSDATA
BS2000
IFG for BERMUDA
User's Guide

[17] TRANSDATA
PDN
TRINIDAD
TRANSDATA Integrated Data Entry Services
User's Guide

[18] TRANSDATA
BS2000
FHS
User's Guide

[19] TRANSDATA
BS2000
DCAM COBOL Calls
User's Guide

[20] TRANSDATA
BS2000
DCAM Program Interfaces
Reference Manual

[21] TRANSDATA
BS2000
DCAM Macros
User's Guide

[22] TRANSDATA
BS2000
IFG for FHS
User's Guide

[23] TRANSDATA
BS2000
TIAM
User's Guide

[24] BS2000
Introductory Guide for
System Users
User's Guide

[25] **Datensichtstation 9750** *)
Betriebsanleitung (Operating Manual)

[26] TRANSDATA
Drucker 9001
Schnittstelle für Programmierer *)
Benutzerhandbuch

[27] TRANSDATA
Drucker 9004
Schnittstelle für Programmierer *)
Benutzerhandbuch

[28] TRANSDATA
BS2000
UTM
Program Interfaces
Reference Manual

[29] TRANSDATA
BS2000
UTM
KDCS COBOL Calls
User's Guide

[30] TRANSDATA
BS2000
UTM
Generation and Administration
User's Guide

[31] TRANSDATA
Datensichtstation 9750 *)
Dialogbetrieb plus BERMUDA
Betriebsanleitung (Operating Manual)

[32] TRANSDATA
BS2000
UTM for RPG Users
User's Guide

[33] TRANSDATA
Datensichtstation 9750-62 *)
Betriebsanleitung (Operating Manual)

*) Available in German only

A

TRANSDATA 9749, 9750, 9752 Data Display Terminals Programmer's Interface User's Guide
Revision November 1984
Order No. U1593-J1-Z76-2-7600

This manual is part of the overall system description of the 7 · 500 / 7 · 700 and TRANSDATA systems. Your comments, as well as any suggestions which you might have, will help us greatly in improving the quality of this manual. Special attention should be paid to those requirements which have practical significance when answering the following questions:

- Where is there too much or not enough detail?
- Where should we provide more supporting diagrams?
- Where is the text difficult to understand?
- Where could the basic structure of the manual be improved?

Please use the back of this page for your suggestions. Give specific page references where appropriate or enclose pages copied from the manual.

Please fill in the following if you require a reply:

From:	
Name	
Company/Department	
Address	
Telephone	Date
Your Siemens office	Mr/Mrs/Miss

Siemens AG
D ST PM 2
Manualredaktion

Otto-Hahn-Ring 6
8000 München 83

Re: Ideas, suggestions, corrections

I am using this manual for

( ) General introduction
( ) Teacher's guide
( ) Student guide
( ) Reference guide
( ) . . . .
. . . . (please state)

I am concerned with:

( ) System operation/administration
( ) System administration
( ) Programming
( ) Systems analysis
( ) Applications engineering
( ) Teaching
( ) . . . .
. . . . (please state)

Page	Remarks

TRANSDATA 9749, 9750, 9752 Data Display Terminals Programmer's Interface User's Guide
Revision November 1984
Order No. U1593-J1-Z76-2-7600

This manual is part of the overall system description of the 7 · 500 / 7 · 700 and TRANSDATA systems. Your comments, as well as any suggestions which you might have, will help us greatly in improving the quality of this manual. Special attention should be paid to those requirements which have practical significance when answering the following questions:

- Where is there too much or not enough detail?
- Where should we provide more supporting diagrams?
- Where is the text difficult to understand?
- Where could the basic structure of the manual be improved?

Please use the back of this page for your suggestions. Give specific page references where appropriate or enclose pages copied from the manual.

Please fill in the following if you require a reply:

From:	
Name	
Company/Department	
Address	
Telephone	Date
Your Siemens office	Mr/Mrs/Miss

Siemens AG
D ST PM 2
Manualredaktion

Otto-Hahn-Ring 6
8000 München 83

Re: Ideas, suggestions, corrections

I am using this manual for

( ) General introduction
( ) Teacher's guide
( ) Student guide
( ) Reference guide
( ) . . . . (please state)

I am concerned with:

( ) System operation/administration
( ) System administration
( ) Programming
( ) Systems analysis
( ) Applications engineering
( ) Teaching
( ) . . . . (please state)

Page	Remarks

TRANSDATA Data Display Terminals 9749, 9750, 9752 Programmer's Interface User's Guide
Edition December 1983
Order No. U1593-J-Z76-1-7600

This manual is part of the overall system description of the 7·500/7·700/4004 and TRANSDATA systems. Your comments, as well as any suggestions which you might have, will help us greatly in improving the quality of this manual. Special attention should be paid to those requirements which have practical significance when answering the following questions:

- Where is there too much or not enough detail?
- Where should we provide more supporting diagrams?
- Where is the text difficult to understand?
- Where could the basic structure of the manual be improved?

Please use the back of this page for your suggestions. Give specific page references where appropriate or enclose pages copied from the manual.

Please fill in the following if you require a reply:

From:	
Name	
Company/Department	
Address	
Telephone	Date
Your Siemens office	Mr/Mrs/Miss

Siemens AG
D ST PM 2
Manualredaktion

Otto-Hahn-Ring 6
8000 München 83

Re: Ideas, suggestions, corrections

I am using this manual for

( ) General introduction
( ) Teacher's guide
( ) Student guide
( ) Reference guide
( ) . . . .
. . . . (please state)

I am concerned with:

( ) System operation/administration
( ) System administration
( ) Programming
( ) Systems analysis
( ) Applications engineering
( ) Teaching
( ) . . . .
. . . . (please state)

Page	Remarks

TRANSDATA Data Display Terminals 9749, 9750, 9752 Programmer's Interface User's Guide
Edition December 1983
Order No. U1593-J-Z76-1-7600

This manual is part of the overall system description of the 7·500/7·700/4004 and TRANSDATA systems. Your comments, as well as any suggestions which you might have, will help us greatly in improving the quality of this manual. Special attention should be paid to those requirements which have practical significance when answering the following questions:

- Where is there too much or not enough detail?
- Where should we provide more supporting diagrams?
- Where is the text difficult to understand?
- Where could the basic structure of the manual be improved?

Please use the back of this page for your suggestions. Give specific page references where appropriate or enclose pages copied from the manual.

Please fill in the following if you require a reply:

From:	
Name	
Company/Department	
Address	
Telephone	Date
Your Siemens office	Mr/Mrs/Miss

Siemens AG
D ST PM 2
Manualredaktion

Otto-Hahn-Ring 6
8000 München 83

Re: Ideas, suggestions, corrections

I am using this manual for

( ) General introduction
( ) Teacher's guide
( ) Student guide
( ) Reference guide
( ) . . . .
. . . . (please state)

I am concerned with:

( ) System operation/administration
( ) System administration
( ) Programming
( ) Systems analysis
( ) Applications engineering
( ) Teaching
( ) . . . .
. . . . (please state)

Page	Remarks

Published by Bereich Datentechnik
Postfach 83 09 51, D-8000 München 83

Siemens Aktiengesellschaft

Published by Bereich Datentechnik
Postfach 83 09 51, D-8000 München 83

Siemens Aktiengesellschaft